Automatic Generation of Dynamic Musical Transitions in Computer Games

Thesis

THE OPEN UNIVERSITY

Automatic Generation of Dynamic Musical Transitions in Computer Games

Simon CUTAJAR

A thesis submitted for the degree of
Doctor of Philosophy

Typeset in LaTeX using TeX Gyre Pagella and PT Serif.

Declaration of Authorship

I, Simon Cutajar, confirm that the research in this dissertation is my own work or that where it has been carried out in collaboration with, or supported by others, that this is duly acknowledged below and my contribution indicated.

I attest that I have exercised reasonable care to ensure that the work is original, and to the best of my knowledge does not break any UK law, infringe any third party's copyright or other Intellectual Property Right, or contain any confidential material.

I confirm that this dissertation has not been previously submitted for the award of a degree by this or any other university.

I understand that my dissertation may be made available to the public, electronically or otherwise.

Details of collaboration

The musical corpus used in my second study (detailed in Chapter 6, and transcribed in the appendix in Section E.3) was commissioned. The entire corpus was composed by Jan Flessel, who was compensated accordingly for his work.

To my mother Isabelle

Acknowledgements

First of all, I'd like to thank my supervisors Dr. Robin Laney and Dr. Alistair Willis. Their help and direction has been invaluable to the shaping of my PhD journey.

Similarly, I'd like to thank Prof. Marian Petre for running the PG Forum, a forum aimed at postgraduate students in the computing department that served to not only bring together students from different disciplines, but also for providing guidance on how exactly one should go about their PhD. Other facilities at the Open University, such as the natural language processing research group, also helped to better focus my research. I'd also like to thank my third party monitor Dr. Soraya Kouadri for her advice, as well as Dr. Allan Jones, Dr. Robert Samuels, and Dr. Alex Kolassa for interesting comments and discussions about my research. Finally, I would also like to thank my viva examiners Dr. Colin Johnson and Dr. Ben Winters.

My friends, both old and new, have all helped to support me during my PhD by providing feedback, suggestions, and being there to listen to my venting when things went wrong. I'd like to thank Aaron, Allan, Amel, Anders, Andrea F., Andrea M., Angelo, Antoniette, Barbara, Bernard, Christina, Eiman, Emilie, Enrico, Germán, Giorgio, Giulia, Maria, Kris, Kristín, Jaisha, Julian G., Julian Z., Matteo, Matthew, Paco, Patrizia, Pinelopi, Sarah, Suraj, Theodoros, Tina, Tracie, William, Yannick, and many more for all the encouragement and support. I'd also like to thank all my study participants for their interest and for taking the time to get involved in my research.

I'd also like to thank my Dungeons & Dragons group for all the adventures we had together, and for being such wonderful players in the world I had built. I would also like to thank the house band at the KMi department at the Open University: the Knowledge Media Instruments, for taking me on as a singer and percussionist. Both groups served to make sure that I had hobbies that took my mind off the PhD.

My amazing girlfriend, Stef, has been a stable element in an often wildly unpredictable PhD. I am ever grateful for her constant love, patience, and support.

Last but not least, I'm grateful to my family for encouraging me to pursue this PhD, for providing useful advice and feedback, and for being supportive in every way. I'd like to thank my mother Isabelle, my brother Stephen, and my first cousins once removed Godwin and Adrian.

This page intentionally left blank.

Abstract

In video games, music must often change quickly from one piece to another due to player interaction, such as when moving between different areas. This quick change in music can often sound jarring if the two pieces are very different from each other. Several transition techniques have been used in industry such as the abrupt cut transition, crossfading, horizontal resequencing and vertical reorchestration among others. However, while several claims are made about their effectiveness (or lack thereof), none of these have been experimentally tested.

To investigate how effective each transition technique is, this dissertation empirically evaluates each technique in a study informed by music psychology. This is done based on several features identified as being important for successful transitions. The obtained results led to a novel approach to musical transitions in video games by investigating the use of a multiple viewpoint system, with viewpoints being modelled using Markov models. This algorithm allowed the seamless generation of music that could serve as a transition between two composed pieces of music. While transitions in games normally occur over a zone boundary, the algorithm presented in this dissertation takes place over a transition region, giving the generated music enough time to transition.

This novel approach was evaluated in a bespoke video game environment, where participants navigated through several pairs of different game environments and rated the resulting musical transitions. The results indicate that the generated transitions perform as well as crossfading, a technique commonly used in the industry. Since crossfading is not always appropriate, being able to use generated transitions gives composers another tool in their toolbox. Furthermore, the principled approach taken opens up avenues for further research.

Contents

List of Tables

List of Figures

List of Musical Examples

List of Acronyms

FPS first-person shooter. 5, 11, 19, 23, 35, 37, 165, 191

JRPG Japanese role-playing game. 4, 59, 62, 87, 169, 170, 174

LARP live action role-playing game. 17

MIDI Musical Instrument Digital Interface. 24, 25, 46, 54, 58, 73, 89, 96, 133, 165, 179, 180, 196

MMORPG massive multiplayer online role-playing game. 5, 6, 17, 69, 73, 172, 175

PCG procedural content generation. 25, 27, 34–37

RPG role-playing game. 13, 19, 66, 88, 169

TTRPG tabletop role-playing game. 22, 34, 68

Games are both object and process; they can't be read as texts or listened to as music, they must be played. Playing is integral, not coincidental like the appreciative reader or listener.

Espen Aarseth, "Computer Game Studies, Year One"[1,2]

1 Introduction

Much has changed since the development of *Pong* (Atari 1972), considered to be one of the first video games. Advances in video game technology have resulted in hyper-realistic graphics, advanced AI and physics engines, and innovative gameplay mechanics. Furthermore, games are growing larger in size and scope, requiring large teams of people in order to build and create the amount of content necessary for them to be released. One such solution to this issue is by using algorithms that are able to quickly create such

[1]Espen Aarseth (2001). "Computer Game Studies, Year One". In: *Game Studies* 1.1. URL: `http://gamestudies.org/0101/editorial.html`

[2]Throughout my PhD, I was inspired by wider reading on the topic. As acknowledgement of my intellectual debt to those authors, each chapter is prefaced with a quote that meant something to me while writing. A footnote with a full citation is provided for each quote.

content. However, as stated by K. Collins (2009, p. 12), there are several reasons why this has not become more common for music in games. One such reason is the requirement of computationally expensive algorithms in order to generate such music, which must also compete for resources with the rendering of graphics or the processing of complicated AI techniques.

1.1 Problem Area

Music plays a large part in setting the mood and atmosphere in digital games by working together with the narrative and complementing the gameplay (Herzfeld 2013). In both games and films, music is used to heighten tension in stressful situations or to emphasise happiness in joyful situations. However, films are a linear medium that have a defined start and end, and watching a film again does not change the narrative or the order in which events takes place. This is not the case for most computer games. Since the player has an active role in progressing through the game, the amount of time that a player takes to experience certain parts of the game and the order in which the game's narrative can be experienced may differ from player to player. This non-linearity means that it is impossible to know how long different game states will be experienced for.

One particular issue that presents itself when composing music for video games is how long the music should be for each game state, making it problematic when trying to successfully transition between two distinct pieces of music. In essence, this problem is due to adding music (traditionally a linear medium) to video games (a non-linear medium). A common solution to this problem is to use the crossfading technique between the music pieces, i.e. lowering the volume of the first piece, while raising the volume of the second piece. However, pieces of different genres, tempo, or simply containing a different rhythm cannot use this technique easily, since these mismatches can cause the result to contain clashing notes or a rhythm that does not match; Berndt (2009, p. 356) refers to these audible clashes as "unintended disharmony and rhythmic stumbling".

This dissertation investigates the use of algorithms to automatically generate music for games. In particular, it aims to generate dynamic musical transitions between two pieces of precomposed music. This will allow game composers to create music that they feel is fitting for particular scenarios or game environments while allowing the algorithm to handle the nonlinearity and unpredictability of when to begin a musical transition in response to player choices.

1.2 Research Question and Objectives

The research question that is being considered in this dissertation is as follows:

> *How can suitable musical transitions be generated automatically within the context of a digital game?*[3]

This main question can be broken down into several different objectives, as described below. Each of these objectives will be tackled during the course of this dissertation.

- Investigate the criteria of success for a musical transition.
- Investigate the use of a predictive technique – multiple viewpoint systems – to generate musical transitions.
- Develop an algorithm that is able to transition successfully between two pieces of music in the context of a video game.
- Evaluate the algorithm based on its ability to keep players immersed.

[3]While this dissertation's main focus is video games, there are some similarities with other types of interactive media such as interactive theatre and interactive film. Aspromallis and Gold (2016) define these as being Extended Performance Experiences.

1.3 Clarification of Scope

First of all, this dissertation focuses on the generation of music in the Western tradition. This is because a large amount of the prior research on generating music also focuses on Western music. Furthermore, most game composers tend to use the Western style of composition when creating music for video games.[4] While the results of this research may very well be applicable to other music traditions, this will not be discussed in this dissertation.

The main focus of this research is its applicability to digital games, which Juul (2005, p. viii) characterises as "games played using computer power, where the computer upholds the rules of the game and where the game is played using a video display". More specifically, this dissertation focuses on open-world games, where players are allowed to explore large expanses of the game world due to the large variety of music associated with different areas in such a world.

Finally, this dissertation will be focusing on the generation of in-game music, which is defined by McAlpine et al. (2009, p. 97) as being "music which accompanies the player's interaction with the main gameplay elements". This is in contrast with other types of music commonly used in games, such as cutscene music which does not offer the player any level of interaction and is therefore more similar to traditional film music.

1.4 Video Games as a Primary Source

When studying music systems in games, it is particularly important to play the game in question and interact with the music system. This allows researchers to understand how the system works within the context of a video game. Indeed, Pinchbeck et al. (2009)

[4]This is also true for many popular non-Western video games, such as Japanese role-playing games (JRPGs) in the *Final Fantasy* series. One counter-example of a game that makes heavy use of traditional music is the action/adventure game *Ōkami* (Clover Studio 2006).

states that it is “highly questionable . . . to not actually engage with the primary data”. However, some games are no longer legally available to play or purchase for a variety of reasons.

For example, some games are no longer available due to the developers or the publishing company having gone out of business, licensing agreements having expired, or the ownership of such rights being unclear (such as with the first-person shooter (FPS) *No One Lives Forever* (Monolith Productions 2000), which is in a state of legal limbo as it is currently unclear as to who owns the rights to the game.). Other games can only be played online or their design requires a purely multiplayer experience, such as massive multiplayer online role-playing games (MMORPGs), and rely on central servers that are often hosted by the game developers. If these servers are shut down, or the game is discontinued (such as with *Asheron's Call* (Turbine Entertainment Software 1999)) the game can no longer be played. Games that have been released on consoles or other hardware that is no longer available can also no longer be played without owning the necessary hardware.

The Digital Millennium Copyright Act (DMCA), specifically Section 1201, is a US copyright law that further complicates this issue by making it illegal for anyone to circumvent copyright protection even if this falls under fair use, hindering researchers looking to study games that are no longer available due to the aforementioned reasons. Museums and libraries were granted an exemption to this law in 2015 (McSherry et al. 2015) in order to preserve games that have been abandoned, but this exemption was only recently extended to include online games (Hansrajh 2018).

One possible solution to this issue is emulation, where “the environment used to run the game in its original format is recreated virtually on a contemporary platform” (Pinchbeck et al. 2009). However, this presents its own set of issues and challenges, such as the authenticity of all aspects of emulated experience, such as gameplay and music. For example, K. Collins (2008, p. 173, footnote 2) states that one emulator she encountered actually caused the audio to be played back half a semitone off.

Furthermore, as stated by Summers (2016, p. 35), while gameplay recordings may be found on YouTube, this does not allow researchers to fully experience the game for themselves, particularly when trying to study how music reacts to different in-game events. This is particularly tricky when researching games that can no longer be played (such as abandoned MMORPGs, which no longer have the player base necessary to recreate the in-game experience), since the researcher can no longer interact with the game, missing out on the experience of the primary data source.

It is also important to specific which version of the game is being discussed due to the possibility of differences in music between versions. For example, Summers states that releases of games in different geographical regions have different music to better suit their intended audience (such as Japanese releases of the racing game *Gran Turismo* (Polys Entertainment 1997) having different music from Western releases (Summers 2016, p. 26)). Furthermore, remasters or re-releases of a game may also include changes to its music (such as with the 2009 remake of *Monkey Island 2: LeChuck's Revenge* (LucasArts 1991), and with the 2002 remake of *Resident Evil* (Capcom 1996), amongst other examples (Summers 2016, p. 27)).

In a similar manner to Summers (2016, p. 231), and in an attempt to take into account the aforementioned issues, I have listed which version of the game has been used in this dissertation in Appendix A.

1.5 Main Contributions

The main contributions of this dissertation can be summarised as follows:

- **identifying suitable features that can be used to evaluate transitions**

  - Musical transitions have unfortunately not been studied in as much depth as they deserve, meaning that not much research has been done in how to evaluate them (both human-composed and otherwise). In Chapter 2, this dissertation looks at several features that have been identified as being essential to a good transition from disciplines such as music theory, ludomusicology, and film studies among others.

- **empirically evaluating the success of different transition techniques**

  - While several features have been described as being essential for a good musical transition, these have not been empirically evaluated in any way. In Chapter 3, this dissertation conducts a study that adapts methods from music psychology in order to evaluate the success of four different musical transition techniques. The results from this study are evaluated in Chapter 4, which takes a look at which transition techniques were considered successful by participants and the features identified as being a part of their success.

- **reformulating the transition problem, and constructing a novel transition algorithm**

  - In Chapter 5, this dissertation discusses the limitations of common implementations of musical transitions in video games, and develops an approach that may be used to allow generative music transitions. A novel transition algorithm that makes use of Markov models and multiple viewpoint systems is also described in this chapter.

- **evaluating the new transition algorithm within the context of a video game environment**

  - In Chapter 6, the newly proposed transition algorithm is placed within the context of a video game environment and compared to the crossfading transition technique which is commonly used in the industry, while in Chapter 7, the results of this study are evaluated based on the identified features described in Chapters 2 and 3.

1.6 Dissertation Overview

This chapter has introduced the main ideas and context of this dissertation.

Chapter 2 further places this dissertation into context by investigating related work that has been done in previously conducted research. This chapter first takes a look at how and why music is used in games, and the role that film music played in its development. Secondly, several algorithmic techniques that have been used in music generation, as well as content generation techniques used in video games, are also discussed. Finally, transitions and other related techniques are discussed both in the context of music theory, as well as video games.

Chapters 3 discusses a study involving the detection of various different musical transitions. Chapter 4 analyses the results of said study, presenting results that inform the development of a new transition algorithm.

Chapter 5 discusses a novel approach that is taken in order to solve issues of abruptness with traditional techniques. This chapter also discusses the design of a novel transition algorithm, which is based on the use of Markov models and a multiple viewpoint system.

Chapters 6 places this new transition algorithm within the context of a digital game environment in order to evaluate its success and the level of immersion that players experience. Chapter 7 analyses said results in order to examine whether or not such a transition algorithm has achieved the objectives of this dissertation.

Chapter 8 concludes the dissertation by revisiting the research question and discussing the main conclusions acquired from the research, as well as possible next steps that research could take based on the work conducted here.

Just as those studying film must engage with films as critical viewers, so we must engage with video games as critical players. We need to be both players and analysts of the games we discuss; it is through playing, listening and interacting that we come to know and understand the music we research.

Tim Summers, *Understanding Video Game Music*, pp. 34–35[1]

2

Literature Review

This chapter discusses previous work on automatically generating musical transitions for video games. Section 2.1 discusses why music is used in games, and the role that film music has played in its development. Section 2.2 discusses various techniques that have been used for algorithmic composition. Finally, Section 2.3 looks at different techniques used for musical transitions and their use in games.

[1] Tim Summers (2016). *Understanding Video Game Music*. With a forew. by James Hannigan. Cambridge University Press. ISBN: 9781107116870. DOI: 10.1017/CBO9781316337851, pp. 34–35

2.1 Music in Video Games

The video game industry has become one of the biggest industries in the world, accounting for more than half of the entertainment market in the UK in 2018 (where it was larger than both the music industry and the film industry combined) (Entertainment Retailers Association 2019).

Both music artists and record labels are becoming more involved with video games and collaborating with game publishers by contributing music to a game's soundtrack (Tessler 2008), and sometimes entire game soundtracks are made up of popular music (such as in the action/adventure game *Grand Theft Auto V* (Rockstar North 2013)) and the top-down shooter *Hotline Miami* (Dennaton Games 2012). Video game music being performed by national orchestras and played in concert halls[2] is becoming increasingly common, and pop artists themselves have been influenced by video games (Muriel and G. Crawford 2018, pp. 46–47). It has also been represented at the Grammy Awards: *Civilization IV*'s (Firaxis Games 2005) theme song *Baba Yetu*, written by Christopher Tin, was the first piece from a video game to be nominated for and to win a Grammy award, and was awarded *Best Instrumental Arrangement Accompanying Vocalist* at the 2010 Grammy Awards. Austin Wintory's soundtrack to the adventure game *Journey* (thatgamecompany 2012), was also nominated for *Best Score Soundtrack for Visual Media* at the 2013 Grammy Awards. Not only does the appearance of video game music demonstrate its impact on popular culture, it also demonstrates its growing acceptance as an art form.

Throughout the literature, a distinction is made between the terms *music* and *sound effects* in games. The term *sound effects* is defined as "non-musical sound" (K. Collins 2008, p. 165), and covers sounds made by objects in the game (such as gun shots, explosions,

[2]Such examples include the *Playstation in Concert* 2018 event performed by the Royal Philharmonic Orchestra at the Royal Albert Hall in London, as well as the *Distant Worlds* 2018 event performed by the Distant Worlds Philharmonic Orchestra at Carnegie Hall in New York City, USA.

and foot steps) as well as sounds made by the game's user interface. *Music* refers to any in-game music that accompanies the player during gameplay. This dissertation will focus purely on music.

2.1.1 Definitions

There is no widely accepted definition for the term *game*. While there have been many attempts to define games, there are always counter-examples that seem to break these definitions. Wittgenstein (1986, pp. 31–32) attempted to search for similarities between different games such as board games, card games, folk games, and sports. He came to the conclusion that while there is no set of features that are common to all games, there are "family resemblances" between different categories of games.

C. Crawford (1984, pp. 2–12) provides a taxonomy of games, paraphrased by Bateman (2016, p. 74) as follows: a game is "something that was created for money, is interactive, has a defined goal, has one or more agents to compete against, and has the means to impede such opponents". However, as stated by Juul (2005), Suits (1978, p. 41) demonstrates that "the simplest way to test a game definition is to test it for being either too broad or too narrow", and so this definition falls short since several games may be identified that do not fit within this definition. For example, *Minecraft* (Mojang 2011) is an open-world sandbox game where players can manipulate the environment by mining and building, but which does not specifically contain a defined goal. Moreover, several games are created with no intention to make money from the results; one such example is the FPS *.kkrieger* (.theprodukkt 2004) made by demoscene[3] programmers in order to create a game that fits within 96kb. Bateman (2016, p. 74) cites the online game *The Endless Forest* (Tale of Tales 2006) as being one example that doesn't meet any of Crawford's requirements, except for it being interactive.

[3]The demoscene is a subculture within the realm of computing where programmers write short programs called *demos* that show off their programming and visual skills.

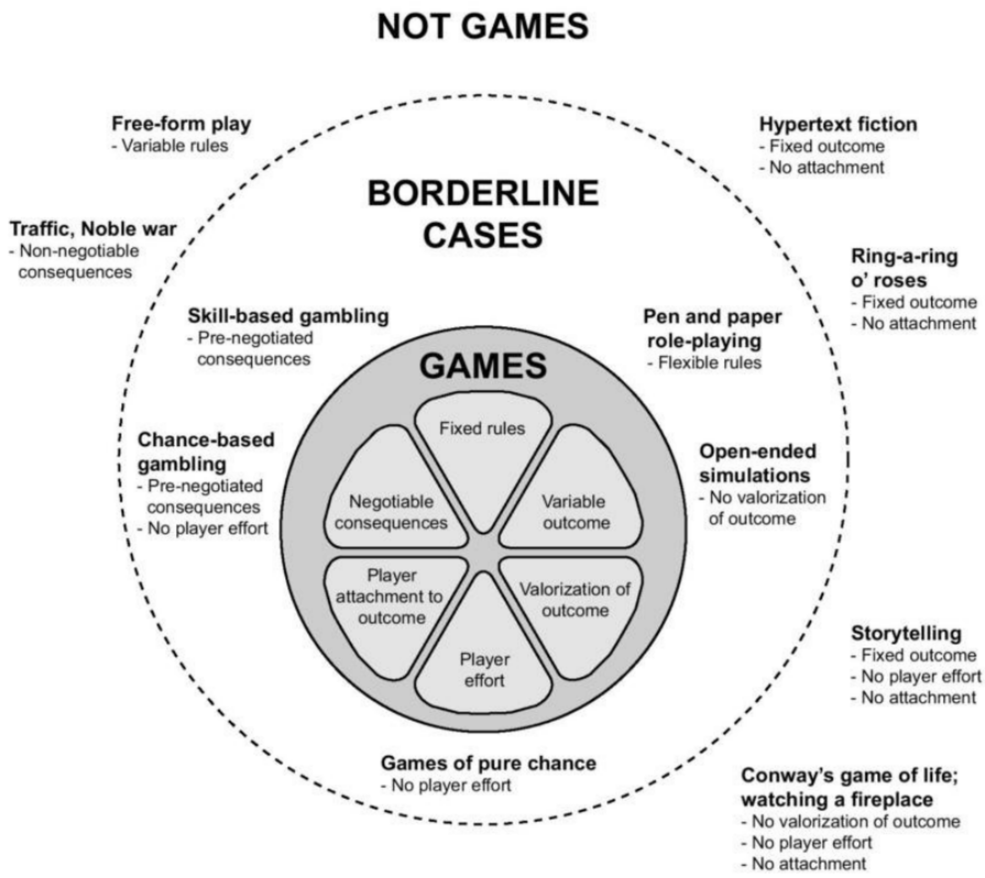


Figure 2.1: Juul's game model, taken from Juul (2005, p. 44)

Fig. 2.1 illustrates Juul's (2005, p. 44) game model, which presents examples of what Juul considers to be *games*, *borderline cases*, and *not games* in an attempt to provide a definition for *game*. In particular, Juul states that games are rule-based, have different outcomes that can be measured, have outcomes that can be classified as positive or negative, require players to work towards an outcome, enable players to be emotionally invested in the outcome, and can be played "with or without real-life consequences". Here again, several games do not fit within this definition; the text adventure game *Colossal Cave Adventure* (Crowther and Woods 1977) (and all other games within the text adventure and interactive fiction genres) is excluded from Juul's (2005) model due to having a fixed outcome. However, this is not so, as the game has both win states and lose states.

Bateman (2016, p. 72) steps away entirely from attempting to characterise the term *game*, and follows the Wittgensteinian approach of examining various different characterisations of *game*. This is done by extracting aesthetic values being used in order to “reveal a variegated landscape of play, within which all games (whatever this term is intended to mean) find their place”. In particular, Bateman considers the following six aesthetic lenses:

- *problem aesthetic*, which focuses on goals to be achieved or challenges to be solved
- *victory aesthetic*, which focuses on play as a form of puzzle resolution
- *reward aesthetic*, which focuses on the framing of games as rewarding activities, “where the enjoyment comes in part by the accepting of limitations”
- *social aesthetic*, which focuses on diplomacy and socialisation
- *imaginative aesthetic*, which focuses on the “fictional elements of play” such as role-playing
- *uncertainty aesthetic*, which simply focuses on change

Rather than continuing to attempt to define the term *game*, this dissertation will make use of the genres that games are commonly classified as in order to give the reader a better sense of what type of game is being discussed. For example, most game genres (such as action, shooter, and strategy), tend to focus on both the problem aesthetic and the victory aesthetic, while role-playing games (RPGs) tend to focus on the problem aesthetic and the imaginative aesthetic. The genres for the games discussed in this dissertation are taken from websites that review, discuss, and catalogue games like MetaCritic.com and MobyGames.com.

The vast majority of the dissertation will be investigating video games, which Juul (2005, p. viii) characterises as "games played using computer power, where the computer upholds the rules of the game and where the game is played using a video display", which according to K. Collins (2008, p. 3) may include "computer monitors, mobile phones, handheld devices, televisions, or coin-operated arcade consoles".[4]

Music games are excluded within the context of this dissertation, which are defined by Kassabian and Jarman (2016, p. 124) as being games "where the majority of gameplay or activity, and if applicable, winning and losing, are predicated on the ability to make good sound and/or musical choices". Four categories of music games are described by the authors: simulator games such as *Guitar Hero* (Harmonix 2005) where players must press buttons on a physical controller shaped like an instrument in order to stay in time with the music, karaoke games such as *SingStar* (SCEE Studio London 2004) where players sing along to karaoke versions of popular songs, rhythm games such as *Space Channel 5* (United Game Artists 1999) where players must press the correct buttons at the appropriate time to follow the presented rhythm, and "non-games" ranging from digital instruments and apps that allow you to remix music.

While the games excluded by Kassabian and Jarman's (2016) definition only involve games that deal with the production of music as their main focus, this dissertation also excludes games where music is tied to the game mechanics in a significant way. Some examples of games that are excluded include *Otocky* (SEDIC 1987) which is a side-scrolling shooter where the player's actions create the game's melody, *Electroplankton* (Indieszero 2005) which is a musical toy allowing players to control various different plankton in order to create music, games where levels are created depending on the

[4] While the majority of this dissertation discusses the music in video games, the title of this dissertation is *Automatic Generation of Dynamic Music Transitions in Computer Games*, where according to the definitions provided by Juul and K. Collins, computer games are a subset of video games. While it is expected that the techniques discussed in this dissertation will work without issues on other platforms such as mobile devices and consoles, they were only tested on a desktop computer and may require additional refinements to work on such platforms.

associated music such as the rhythm games *Dance Dance Revolution* (Konami 1997) (where players must step on areas of a dance mat that correspond to a pre-designed sequence of symbols on screen at the right time, in effect making the player dance to the music) and *Audiosurf* (Fitterer 2008) (where players must navigate a track by collecting blocks on separate lanes; this track is automatically generated depending on the chosen music), and rhythm games that use other mechanics such as *Crypt of the NecroDancer* (Brace Yourself Games 2015) and *Cadence of Hyrule* (Brace Yourself Games 2019), both of which are roguelike[5] rhythm games where players must collect items and defeat enemies while moving the character in time to the music.

In a similar manner to games, there is also no widely accepted definition for the term *music*. A commonly used definition is one given by Edgard Varèse; music is "organized sound" (Goldman 1961, p. 133). However, this is too broad a definition, as human speech certainly fits within this definition, but does not seem to be regarded as music. Since defining *music* falls outside the scope of this dissertation, the following characterisation by Kania (2011) will be used:

> Music is (1) any event intentionally produced or organized (2) to be heard, and (3) *either* (a) to have some basic musical feature, such as pitch or rhythm, *or* (b) to be listened to for such features.[6] (Kania 2011, p. 12)

In the context of this dissertation, the above serves to encompass both human-composed music as well as algorithmically generated music, and game music that spans a variety of different genres.

[5]The term *roguelike* is used to describe games that are similar to the game *Rogue* (Toy et al. 1980), but is not particularly descriptive to anyone who hasn't played such games. As stated by Garda (2013), the *Berlin Interpretation* is a series of factors that characterise such games, and was developed by the roguelike community at a development conference in Berlin. These factors include random environment generation, permanent death, grid-based gameplay, and tile-based gameplay among other factors.

[6]Kania discusses how the circular definition is avoided by defining a musical feature as a sound intended to be musical, relying on the musician's intention rather than the perception of the listener.

2.1.2 Similarities Between Film Music and Game Music

To understand why music is used in games at all, a brief comparison is made to the use of music in films. Sabaneev (1935, p. 18) states that music and cinema were always linked from the very beginning, such as by having pianists play along in real time to the film being screened. In this way, the pianist's "aesthetic function was…to fill up the tonal void which was an inherent feature of the silent film". This is similar to claims made by Berndt and Hartmann (2008, p. 126), who state that the "tremendous importance [of music in games] only becomes apparent when taking it away, i.e., muting the music, and experiencing the disillusioning lack of this unconscious something". Similarly, Lankoski (2012, p. 40) states that "[o]ne needs only to turn off the sounds from the *Silent Hill 3* (Konami Computer Entertainment Tokyo 2003) or *Thief: Deadly Shadows* (Ion Storm 2004) to see how the games lose their emotional impact".

In a similar manner to film, music in games is used to accompany the visuals and to set the tone and mood of the scene. Therefore, not only is the use of music in games important, "[it] is accepted, even expected, because videogame players' expectations are often informed, inter-medially, by filmic convention" (Hooper 2016, p. 125). Similarly, K. Collins (2008, p. 137) states that "the realism aspired to in games is not a naturalistic realism in the sense of being a simulation of reality, but a cinematic realism that relies on established motion-picture convention". Hooper gives an example of this for the game *Red Dead Redemption* (Rockstar San Diego 2010), a Western-themed action-adventure game that instead of using period music as background music, relies on musical themes similar to Ennio Morricone's music composed for Sergio Leone's Western films in the 1960s. In contrast, K. Collins (2013, p. 3) states that the musical accompaniment used in certain games is closer in style to cartoons, "with music more intimately tied to action, rhythm, and movement and stylized in ways that are often not intended to be realistic"; this technique is often called *Mickey-Mousing* and is a reference to Disney and Iwerks's

(1928) short film *Steamboat Willie*, where the character Mickey Mouse was first introduced and where this technique first appeared.

However, there are also some differences in how music works in games, and how music works in film. Sweet (2014, pp. 16–20) describes five differences between the two:

- *differences in interaction*: while watching a film is passive, meaning that there is no interaction between the film and the viewer other than following the story, a game requires active interaction for players to reach the conclusion.
- *structure*: a film is normally linear and contains "one beginning, one middle, one end" (Sweet 2014, p. 16), while a game is nonlinear and may contain multiple endings or alternate paths for players to explore.
- *variable length*: films have a fixed length, with the average length of a film being between 1 and 3 hours, while the average length of a game depends the type of game and its genre. Sweet gives three examples of game types: casual games which have an average play time of 2-3 hours, console games which have an average play time of 10+ hours, and MMORPGs which have a total play time of 50+ hours.[7]
- *multiple sessions*: while a film is normally watched in its entirety in one sitting, most games are played over several sessions due to their sheer length. This means that each gameplay session will itself have its own individual soundtrack consisting of those parts and sequences of a game's music heard during a session, and that "[t]he totality of the game's music is [...] not encompassed by one instance of the game being played"[8] (Summers 2016, p. 25).

[7]While not a video game example, Summanen (2004) describes the process of composing music for the live action role-playing game (LARP) *Mellan himmel och hav* (Wieslander and Björk 2003). In total, Summanen composed 72 hours worth of music for the game, an approach that is not practical for video game studios (both in terms of storage space, as well as production time and effort) and would not take into consideration any of the issues being discussed here.

[8]While TV series and other episodic media are not normally consumed in one sitting, each episode contains its own self contained soundtrack that does not change with repeated viewings. This is in contrast

- *amount of music*: the amount of music composed for a film is slightly shorter than or about the same length as the film. In contrast, games tend to be much longer than films and therefore require much more music. However, most games tend to resort to looping or reusing music in different in-game locations, since it is impossible to know just how much music needs to be composed due to a game's variable length.

Sweeney (2014, p. 37) claims that the terminology used in the analysis of video game music is derived from that of film music, with more game-specific terminology being used as the area progressed. However, K. Collins (2007) claims that comparisons between film music and game music have resulted in the implication that games are inferior to film, making reference to Bessell (2002, p. 142) who suggests that techniques used by Boulez should be used when composing music for games, unaware that these techniques had already been used since the early 1980s.

2.1.3 Music's Role in Video Games

As mentioned previously in Section 2.1.2, a large amount of terminology used in the analysis of video game music has its origins in film studies. One such example is the distinction between *diegetic* audio and *extra-diegetic* audio, which can apply to both music and sound effects in games.

diegetic audio
: audio that occurs within the game itself and can therefore be heard by the player character (such as the sound of footsteps being created by the player running through the level, or music being played by the bard in the tavern that the player character has just entered)

extra-diegetic audio
: audio that occurs outside the game and can not be heard by the player character

to a player's interaction with a video game, where each gameplay session may have a different soundtrack each time depending on the player's actions within the game.

(such as a menu selection sound in a menu screen, or music that is heard at the end of a level in a game)

One game that makes use of both diegetic and extra-diegetic music is *Bioshock* (2K Boston and 2K Australia 2011), a dystopian FPS set in the 1960s and set in an underwater complex. Gibbons (2011) states that *Bioshock* makes use of diegetic music by playing music from the 1930s, 1940s and 1950s through speakers in the in-game underwater complex that the player character can hear, but also makes use of extra-diegetic music by providing orchestral music during battles.

However, this distinction between diegetic and extra-diegetic music is not without issues. Winters (2010) states that while these terms originated from literary theory and were sufficiently descriptive there, applying them to film music assumes that music plays a limited role in the film's narrative, which is not the case. In addition, Jørgensen (2010) states that game music has additional roles that are normally not present in film music. Such roles include usability, where the music helps inform players on how to progress in the game, and informs gameplay by guiding players in different scenarios.

Due to these limitations, Jørgensen (2007, p. 105) introduces the term *trans-diegesis*. Here, trans-diegetic music is defined as being "[m]usic with no source in the game world but still [having] the ability to inform about events in that world". While most uses of the term trans-diegesis is with respect to sound effects in games, Jørgensen (2008b) gives the example of the use of combat music in the action/RPG game *The Elder Scrolls IV: Oblivion* (Bethesda Game Studios 2006) which starts playing as soon as an enemy is close to the player character. This allows players to prepare for combat even though they might not have even seen the enemy.

In summary, music plays various different roles in video games. Zehnder and Lipscomb (2006) state that music can provide a sense of immersion, complement the game narrative, or provide emotional subtext to a scene. In addition, as stated by Jørgensen (2010), game music can help inform players about their progress, as well as inform gameplay.

2.1.3.1 Different Uses of Game Music

While the previous section highlighted several different roles that music plays in video games, these roles vary in prominence depending on the genre of video game in question. Sweet (2014, pp. 329–330) describes three possible categories of games and the different ways in which they use music:

closed systems (*games on rails*)

Sweet claims that games that fall into this category, such as *Uncharted: Drake's Fortune* (Naughty Dog 2007) or *King's Quest 6* (Sierra On-Line 1992), are very heavily story driven, and are normally split up into self-contained segments separated by checkpoints. These checkpoints can provide some indication to the audio engine that the music should change. Sweet states that although these type of games excel at "emotionally connecting the player to the story", they are generally unable to react to different emotional states within the segment.

open systems (*open-world games*)

Games that fall into this category, such as *World of Warcraft* (Blizzard Entertainment 2004), allow the players to explore a large open world. Sweet states that music is composed to distinguish between different zones in the world, and the music may not necessarily change when different events are encountered (such as encountering enemies).

combat-triggered music interactivity

Sweet states that many games make use of music to indicate whether a player has entered combat; examples can be seen in *Final Fantasy X* (Square 2001) or *Pokémon Red* (Game Freak 1996) where combat is presented as random encounters that occur while walking around in specific areas on a world map. However, Sweet claims that most music engines do not cater for any differences that may occur during combat (such as whether the player is close to death, or close to victory).

McAlpine et al. (2009) also distinguish between different types of game music, which take on different roles depending on which part of the game is being experienced:

title music
: the first piece of music that the player hears when loading the game, or at the title screen; this is normally used to set the overall mood. Munday (2007) also claims that this type of music serves as bridge between the real world and the virtual world.

menu or high score music
: referred to as the "elevator music of the video game world" by McAlpine et al., this type of music is low-key and used simply to prevent silence.

cutscene music
: accompanies an in-game cutscene, which is a short sequence where players have no interaction and simply watch a sequence of events play out. Since cutscenes are linear, the music that accompanies them tends to be linear and works similarly to film music[9]

in-game music
: as described by McAlpine et al., this is "music which accompanies the player's

[9] For a more in-depth analysis of the use of music in video game cutscenes, see Hooper (2016).

> interaction with the main gameplay elements".[10] In particular, Stockburger (2003, pp. 6–7) states that music can be tied to different locations in the game, increasing the player's spatial experience of the game.

Following these definitions, it can be seen that *in-game music* is the most likely to change depending on the player's interaction in the game. Cutscenes are normally linear and have a fixed length, so cutscene music does not need to be interactive as they can be composed to fit the scene precisely (Whalen 2004). Menu music may similarly be influenced by the player's interaction with those screens, although there are fewer opportunities to provide this interaction. The title screen contains little to no interactive elements, so *title music* does not need to be interactive either (although a composer can never know how long the player is willing to remain on the title screen).

The focus of this dissertation will be on the applicability of generative techniques to in-game music, since both menu music and hi-score music provide very little interaction in comparison. Furthermore, the player spends the vast majority of the game experience actually playing the game, rather than in the menus or the hi-score screens.

2.1.3.2 Game Music and Immersion

There seems to be no clear and unified definition for the term *immersion* in the literature; indeed, E. Brown and Cairns (2004) claim that the term's definition changes depending on the underlying motivations and its context, while Calleja (2011) states that immersion has been applied to different types of media (such as paintings or text), which provide a

[10]While this dissertation is about video game music, there are some parallels with how music is used in tabletop games. Jamison (2009, pp. 222–223) discusses how music can be used effectively in tabletop role-playing games (TTRPGs) by catering to the appropriate mood, and several music albums have been composed specifically to be used in a TTRPG setting, such as Midnight Syndicate's album *Dungeons and Dragons* (Midnight Syndicate 2003). Similar tools exist for board games, such as the website *Melodice* (`http://www.melodice.org`), which provides crowdsourced playlists that are appropriate during gameplay. Furthermore, some tabletop games even release their own selected background music; the mobile app *One Night* (Bézier Games 2013) automatically directs the game and serves as accompaniment to the hidden identity party game *One Night Ultimate Werewolf* (Alspach and Okui 2014).

different experience to games. Furthermore, the term has been associated with experiences as varied as "general engagement, perception of realism, addiction, suspension of disbelief, [and] identification with game characters" by both industry and academia, contributing to the confusion around the term's definition (Calleja 2011, p. 25).

Sanders and Cairns (2010) distinguish between the terms *immersion*, *flow*, and *presence*. Although they are frequently used interchangeably in the literature, the authors claim that these terms have clear definitions and refer to slightly different concepts. Sanders and Cairns, p. 160 define *presence* as "the sense a person has of actually being located in the virtual space" in an environment in virtual reality or games using a first-person view. The authors claim that players may feel present but not immersed in a particular experience, and gives the example of a player performing a boring task in a realistic virtual environment. *Flow* is defined by Sanders and Cairns (2010, p. 160) as "an extreme experience where goals, challenges and skill converge", and Nakamura and Csikszentmihalyi (2002, p. 90) state that there are several conditions that must be fulfilled before entering a state of flow, such as "intense and focused concentration on what one is doing in the present moment", "loss of reflective self-consciousness", and "distortion of temporal experience" amongst others. While Sanders and Cairns claim that flow is a binary experience, immersion is contrasted with flow by being able to be measured along different levels of intensity. Nacke and Lindley (2009) also introduce the concept of *boredom*, which in the context of a game, they describe as being a lack of player engagement that gives the player no experience of immersion or flow.

Work done by Grimshaw et al. (2008) (using an FPS game, and different conditions such as whether or not sound and music were playing), Gormanley (2013) (using a hidden object game), and Engström et al. (2015) (using a point-and-click adventure game played by both sighted and blind people) show that music is an important contributor to immersion in video games.

However, there has been very little work done into what specific musical factors contribute towards immersion. For example as stated by both van Elferen (2016) and Hooper (2016), Calleja (2011) only mentions music's role in immersion in passing by using anecdotes. van Elferen (2016) presents a model for analysing game music immersion, made up of three factors: *musical affect* which relates to any personal connection that players might have with the music, *musical literacy* which relates to the knowledge of musical conventions and tropes that are being used, and *musical interaction* which relates to how players interact with the soundtrack. However, van Elferen's (2016) model is purely analytical and cannot be used to evaluate the level of immersion experienced due to the music.

2.1.3.3 Game Music Styles

With a large variety of different game genres and settings, it is understandable that the music should complement the established game world. Video game music cannot be categorised into one musical genre, and one cannot simply say that certain game genres use certain styles of music either. Rather, Munday (2007, p. 51) claims that "today, videogame music inhabits every style imaginable, from baroque to bluegrass, rockabilly to symphonic", and thus is composed depending on the requirements of the intended game.

As such, a variety of different and diverse musical styles are represented in video game soundtracks. Examples include the use of heavy metal and industrial music in the FPS *DOOM* (id Software 2016)[11], jazz in both the run-and-gun platformer *Cuphead* (StudioMDHR 2017) and the point-and-click film-noir adventure game *Grim Fandango* (LucasArts 1998), as well as musique concrète and atonality in the survival horror game *Resident Evil 7: Biohazard* (Capcom 2017).

[11]This version is a remake of the original *DOOM* released in 1993.

Summers (2011) also states that composers take into account both the environmental setting of the game as well as the type of game itself when creating the game's music. While *Civilization IV* (Firaxis Games 2005) and *Dune II* (Westwood Studios 1992) are both strategy games, the different settings in which the games take place change the requirements of the needed soundtrack. In *Civilization IV*, a game where players take a real-world empire from prehistoric times to the modern day, each civilisation has their own leitmotif that makes them distinct from each other, and as the game progresses, that musical theme starts out melodically simple and "[changes] to a version of the theme that is more fully orchestrated, stronger, and more melodically complete" (Summers 2011, p. 13). In contrast, since *Dune II* takes place on alien worlds, the soundtrack shifts from eerily sparse ambience to grandiose space opera. Munday (2007) claims that the type of music in video games can be determined by the amount of narrative content in the game: games with a low narrative content tend to license pop songs that are meant to motivate the player while playing, while games with a higher amount of narrative normally employ composed scores.

2.1.4 History of Music in Games

Pong (Atari 1972) is considered by K. Collins (2008, p. 8) to be one of the first digital games[12] with sound effects in it. This was the beeping sound when the ball collided with the player's paddle, which was implemented entirely within the game's hardware. Music was at first limited to menu screens and game over screens, due to the limitations of the hardware at the time (Fritsch 2013). However, as computers became more powerful and the amount of available memory increased, these limitations started to subside, with K. Collins (2008, p. 12) stating that *Space Invaders* (Taito 1978) "set an important precedent for continuous music, with a descending four-tone loop of marching alien feet that sped up as the game progressed".

[12]While K. Collins (2008, p. 7) states both penny arcades and pinball machines made use of both music and sound effects in order to attract new players, investigating this further is not relevant here.

By the 1980s, dedicated sound chips had been introduced that made use of synthesis, allowing developers greater control over the types of sounds that could be produced. The Musical Instrument Digital Interface (MIDI) protocol was slowly introduced in the mid-1980s, allowing game composers the advantage of relying on instruments built into the sound card, thus saving space within the game. However, this meant that the same piece might be interpreted differently due to the different types of hardware.[13] By the beginning of the 1990s, sound and music were prerecorded onto CDs, allowing game developers to ensure that players would hear the same piece regardless of the hardware they owned, as well as allowing for higher quality audio (K. Collins 2008).

However, the use of high-fidelity sound came with a particular drawback: "the dynamic MIDI techniques had been abandoned in favor of a return to linear tracks and loops" (K. Collins 2008, p. 67). Page and M. Kelly (2007) state that one possibility to remedy this could be the return to MIDI, since "you could have all the [digital signal processors], all the soft synths and everything else actually on the console, and then just play it as a MIDI file". This also has the advantage of allowing the use of real time mixing to create interactive tracks as well as reduced memory usage, with the caveat that the return to MIDI might not be viable for orchestral music. However, K. Collins (2009) states that returning to MIDI might come as a shock to players who are now used to high-quality music in games. This seems to be the case, since little to no modern games have been released that use MIDI. While the approach suggested by Page and M. Kelly (2007) does not seem to have been used much by game developers, the announcement of the MIDI 2.0 protocol may serve to change that (The MIDI Assocation 2019).

[13] The website `http://sound.dosforum.de/` showcases different audio files in order to compare the results of the same pieces of music over different sound cards and synthesisers, for example, to compare the title theme for *The Secret of Monkey Island* (Lucasfilm Games 1990) played using an AdLib soundcard and a Creative SoundBlaster card.

2.2 Algorithmic Composition

This section discusses the history of algorithmic composition, different techniques that have been used to algorithmically compose music, and common uses for these techniques. The term procedural content generation (PCG) is discussed in the context of video games, and different techniques that could be used to generate music for video games are compared and contrasted.

2.2.1 Algorithmic Composition

Several terms have been used to define the concept of automatically composed music that uses little to no input from humans. Ariza (2005) identifies several of these definitions, a few of which include *algorithmic composition*, *computer-aided composition*, *computer music*, and *procedural composition*, while Wooller et al. (2005) also introduce the term *generative music*.

The terms that are the most relevant to the processes used in this research are as follows:

- algorithmic composition
- procedural composition
- generative music
- computer-aided algorithmic composition

Each of these terms will be described in terms of their respective scope and limitations in the hopes of settling on a suitable definition.

2.2.1.1 Algorithmic Composition

Music composition using algorithmic techniques is not a novel technique. Maurer (1999) claims that Pythagoras was one of the first to introduce algorithmic thinking and "formal processes" to the creation of music, while N. Collins (2018, p. 69) states that Guido d'Arezzo's vowel-to-pitch algorithm and the *ars combinatoria* musical dice games were commonly regarded as the first instances of algorithmic composition.

S. A. Hedges (1978) states that the first musical dice games were published in the mid-18th century, and allowed the performers to roll two dice as the piece was being played, generating the reference to the bar of music that should be played next. According to S. A. Hedges, the first published instance of a musical dice game seems to be Kirnberger's (1767) *Der allezeit fertige Polonoisen- und Menuettenkomponist*; this was used as a model for other composers to create and publish similar works. The most frequently cited (and possibly most well known) dice game is *Musikalisches Würfelspiel*, capable of producing waltzes. This was attributed to Mozart (1793), though S. A. Hedges claims that the use of Mozart's name was simply a publicity stunt for the publishers, and that the composer was never really involved.

An interest in algorithmic composition was rekindled with the introduction of electronic computers. Maurer (1999) states that one of the earliest examples of this can be seen in Hiller and Isaacson's (1957) *Iliac Suite*, an algorithmic composition in four movements made by using rule-based systems and Markov models.

2.2.1.2 Procedural Composition

K. Collins (2009, p. 13, see footnote 1) defines procedural composition as being "composition that evolves in real time according to a specific set of rules or control logics". There are two distinct features in this definition; first, the idea that the composition is evolving and changing in real time, and second, that the composition is ultimately being guided by a pre-defined set of rules.[14]

2.2.1.3 Generative Music

Wooller et al. (2005) state that the term *generative music* has been used by scholars to refer to a number of different concepts, as listed below.

linguistic/structural

As stated by Lerdahl and Jackendoff (1996, p. 6): "the sense of "generate" in the term "generative grammar" is not that of an electrical generator that produces electricity, but the mathematical sense, in which it means to describe a (usually infinite) set by finite formal means". In this category, Wooller et al. place music that has been created by theoretical structures inspired by generative grammars.

interactive/behavioural

Here, generative music is defined to mean music that is produced from a non-musical input, such as by using a seed value.

creative/procedural

Wooller et al. group any music "resulting from processes set in motion by the composer" into this category. Note that this is similar in scope to *procedural composition*, as stated earlier in Section 2.2.1.2.

biological/emergent

Music that is non-deterministic, such as wind chimes. Here, Wooller et al. state that this definition has emerged from the term *generative art*.

[14]Note that *procedural* is used as part of the term *procedural content generation* (*PCG*), which is described in more detail in Section 2.2.3

Wooller et al. (2005) attempt to provide a clearer definition by creating a framework used to categorise algorithmic music systems. In this framework, an algorithm is defined as being *generative* "when the resulting data representation has more general musical predisposition than the input and the actual size of the data is increased" when compared to the original input (Wooller et al. 2005). The example given by the authors is an algorithm that takes a seed value as an input and returns a sequence of notes as an output.

2.2.1.4 Computer-Aided Algorithmic Composition

As can be seen, most of the above definitions overlap each other and describe somewhat similar concepts. Futhermore, a detailed investigation of each definition is beyond the scope of this dissertation. However, Ariza (2005) states that a better term that could be introduced is *computer-aided algorithmic composition*, combining the definitions of *computer-aided composition* (which focuses on the idea that composition is aided by the use of a computer, but does not specifically mention the use of generative algorithms. This could essentially describe anything from music notation software to music sequencing software.) and *algorithmic composition* (which does not necessarily mean that a computer is needed).

Ariza (2005, p. 765) states that computer-aided algorithmic composition is "software that facilitates the generation of new music by means other than the manipulation of a direct music representation", where a direct music representation is one that is the same for both the user's input as well as the algorithm's output, as well as "permits the user to manipulate indirect musical representations". However, Ariza admits that this definition is rather vague.

2.2.2 Algorithmic Techniques

There is a vast amount of literature on the different techniques and algorithms that have been used to generate music. In this section, these techniques will be grouped into six different categories:

- grammars
- knowledge-based systems
- Markov models
- artificial neural networks
- evolutionary algorithms
- chaos and self-similarity

This grouping is loosely based on work done by Papadopoulos and G. Wiggins (1999), Nierhaus (2008), and Fernández and Vico (2014). Although I will not be delving into detail on how each technique works, their advantages and disadvantages are discussed with respect to their use in an interactive digital environment such as a video game.

There are several variations as to the scope of the survey papers being referenced here. For example, Fernández and Vico (2014) and Papadopoulos and G. Wiggins (1999) do not include papers that discuss the generation of dynamic and adaptive music (such as in games or other interactive settings). Similarly, Nierhaus (2008) only briefly discusses the use of algorithmically generated music for interactive systems. This is mitigated by Herremans et al. (2017), who include papers that address some of the challenges associated with generating interactive music. Furthermore, while Fernández and Vico (2014) restrict the coverage of their survey paper to algorithmic techniques used for symbolic representations of music, Herremans et al. (2017) have no such restriction.

grammars

Generative grammars are linguistic models based on a finite set of rules. They were first described in detail by Chomsky (1957), with Lerdahl and Jackendoff (1996) applying these techniques to music.

Nierhaus (2008, p. 117) states that one issue with using generative grammars for music generation is "their basic orientation on a sequential model that must neglect simultaneously occurring structure", implying that while suitable for monophony or chord progressions, grammars might not be suitable for polyphonic music. Both Nierhaus (2008, p. 117) and Papadopoulos and G. Wiggins (1999) also state that the grammatical parsing of musical structure is computationally expensive.

knowledge-based systems

This category refers to rule-based systems (or expert systems) which are able to reason through to a problem in order to arrive to a particular conclusion. A prominent example of the use of a rule-based system in algorithmic composition is Ebcioğlu's (1988) CHORAL system, which is able to harmonise melodies in the style of Bach chorales.

Papadopoulos and G. Wiggins (1999) state that rule-based systems suffer from the issue that it is both a difficult and lengthy process to acquire and encode the required knowledge into a system. They also state that the systems are only ever as good as the experts who were involved in building them, and such systems tend to become complicated and convoluted when catering for edge cases.

Markov models

Markov[15] models are stochastic models first described and used by Markov (1924) to model language. By modelling a sequence of events in time, they are able to predict the probability of the next event at a given point in time based on the previous events encountered. Pachet and Roy (2011, p. 149) state that "the Markovian aspects of musical sequences has long been acknowledged" ever since Shannon's (1948) work on information theory, and the earliest examples of these models being used to model music include work done by Pinkerton (1956) and Brooks et al. (1957). Markov models have also been used in conjuction with other techniques, such as in T. Collins's (2011) work on pattern discovery and generation.

According to Nierhaus (2008, p. 81), Markov models are best suited for "one-dimensional symbol sequences" such as melodies. However, melodies can be represented in more complex and effective ways, as shown by T. Collins (2011) and Whorley (2013) who both show that a suitable representation can also allow for the generation of harmony without exponentially increasing the state space. Furthermore, Nierhaus states that they are best suited for style imitation. However, Markov models have several limitations; for example according to Fernández and Vico (2014), low-order models tend to wander aimlessly while higher-order models tend to replicate large chunks of the corpus. However, Fernández and Vico state that these limitations become less apparent in more restricted domain areas, such as real time composition and improvisation and the generation of musical transitions.

[15] As stated by Fernández and Vico (2014), *Markov* and *Markoff* are alternative English transliterations of the Russian surname Марков. While older papers use the transliteration *Markoff*, *Markov* seems to be more commonly used nowadays.

artificial neural networks

Artificial neural networks are models that are inspired by biological neurons. They are made up of an interface layer which takes the input into the system, a propagation function that combines the input values and propagates them to each layer in the model, an activation function which determines the activation state of the model based on a threshold value, and an output function that is able to return the result (Nierhaus 2008, p. 206). There are many different types of neural networks, each with their own strengths and weaknesses, that have been used in algorithmic composition, with a well-known example being Hild et al.'s (1991) HARMONET system which is able to harmonise melodies based on Bach chorales.

Toiviainen (2000, p. 51) states that while neural networks are "capable of successfully capturing the surface structure of a melodic passage and produce new melodies on the basis of the thus acquired knowledge, they mostly fail to pick up the higher-level features of music, such as those related to phrasing or tonal functions".

Deep learning techniques are similarly based on neural network techniques, but involve the use of "multiple layers processing multiple hierarchical levels of abstractions, which are automatically extracted from data" (Briot et al. 2019). Briot et al. state that the main advantage of using deep learning networks is the ability to generalise the model across different musical genres, provided that enough data is provided to the system.

evolutionary algorithms

Evolutionary algorithms are inspired by Darwin's theory of evolution and, according to Nierhaus (2008, p. 158), can be used to search for optimal solutions to problems.

Evaluating the results of an evolutionary algorithm may be done in two ways: automatically by using an evaluation function, or by using human evaluation. Papadopoulos and G. Wiggins (1999) state that the problem with using humans is that the evaluation is subjective; furthermore, both Papadopoulos and G. Wiggins (1999) and Nierhaus (2008, p. 183) state that using a human evaluator may cause bottlenecks, since a large number of candidates would need to be manually evaluated for each generation. Nierhaus (2008, p. 184) states that while evolutionary algorithms may not be well suited for style imitations tasks, they can be used successfully for other compositional approaches such as harmonisation of an existing melody (Fernández and Vico 2014).

chaos and self-similarity

This grouping encompasses several different techniques, including fractals, Lindenmayer systems (also referred to as L-systems), and cellular automata, the latter two of which will be described further.

Originally developed by Lindenmayer (1968) to represent algae growth, L-systems were later applied to music composition by Prusinkiewicz (1986), who used a mapping to convert a graphical representation of an L-system into music. They are frequently grouped together with grammars due to operating in a similar way by using rewriting rules, replacing symbols with a larger string of symbols. Cellular automata are models that use state transition rules on a grid of cells in order to produce a new state. They were first applied to music by Beyls (1989), based on a mapping of note pitches to cells.

According to Fernández and Vico (2014), L-systems have been used successfully when implementing parts of the compositional process, and not the process in its entirety, while Papadopoulos and G. Wiggins (1999, p. 111) state that evaluating the results of these systems is difficult because "unlike

> all the other approaches, their "knowledge" about music is not derived from humans or human works"; this is problematic if the aim of music generation is to further understand a human's compositional process. Finally, Nierhaus (2008, p. 202) states that both L-systems and cellular automata tend to be used for "personal compositional strategies" rather than stylistic imitation.

Based on the techniques discussed above and their applicability to music composition in a real time environment, the rest of this dissertation will focus on the use of Markov models. A further description of how Markov models work and how they may be used in algorithmic composition is given in Chapter 5.

2.2.3 Procedural Content Generation

The use of algorithmic techniques for content generation in games is not a new approach, and has been used in both analogue and digital games.[16] Within the game studies community, the algorithm generation of such content is referred to as *procedural content generation* (*PCG*).

Procedural content generation is defined by Togelius et al. (2011a, p. 6) as "the algorithmical creation of game content with limited or indirect user input". Following Togelius et al.'s (2011b, p. 172) definition for game content: "all aspects of the game that affect gameplay other than nonplayer character (NPC)[17] behavior and the game engine itself". More specifically, the authors state that this definition of game content covers "terrain, maps, levels, stories, dialogue, quests, characters, rule-sets, dynamics, and weapons", while placing less importance on so-called "decorative assets such as lighting, textures, and sound effects" since they do not directly affect gameplay (although there are exceptions for music, such as in music games, as discussed in Section 2.1). Freiknecht

[16] Most notably, these include techniques such as roll tables and modular designs used in TTRPGs such as *Dungeons and Dragons* (Gygax and Arneson 1974). For a detailed look at this topic, see Smith (2015).

[17] A nonplayer character (or NPC) is a character in the game that is not controlled by the player.

and Effelsberg's (2017, p. 5) definition for PCG: "the automatic creation of digital assets for games, simulations or movies based on predefined algorithms and patterns that require a minimal user input" is much more inclusive and less reliant on being relevant to gameplay.

Typically cited examples of the use of PCG algorithms in games all seem to revolve around playable content. The following list presents several examples that are frequently cited in the literature:

- *Rogue* (Toy et al. 1980), a dungeon exploration game that used generated levels for replayability.
- *Elite* (Braben and Bell 1984), a space exploration game that included generated planets that made the in-game universe feel enormous.
- *SpeedTree* (Interactive Data Visualization Inc. 2002), a software tool used by both game developers and film makers alike in order to procedurally generate realistic looking trees.
- *Borderlands* (Gearbox Software 2009), an FPS that uses PCG to generate different types of guns that the players can use.
- *Minecraft* (Mojang 2011), a sandbox game that generates the terrain of a 3D voxel[18] world.
- *Dwarf Fortress* (Adams 2006), a simulation/rogue-like game that generates a world in its entirety, down to its history, geology, and the different cultures that live there.

However, within the literature that discusses PCG in games, very little attention is given to music. In a survey by Hendrikx et al. (2013), PCG algorithms for music

[18] In a similar manner to how a pixel works in a 2D display, a *voxel* represents a value in a 3D grid. In *Minecraft*, these voxels are represented by using differently textured blocks that can be used to build virtual worlds.

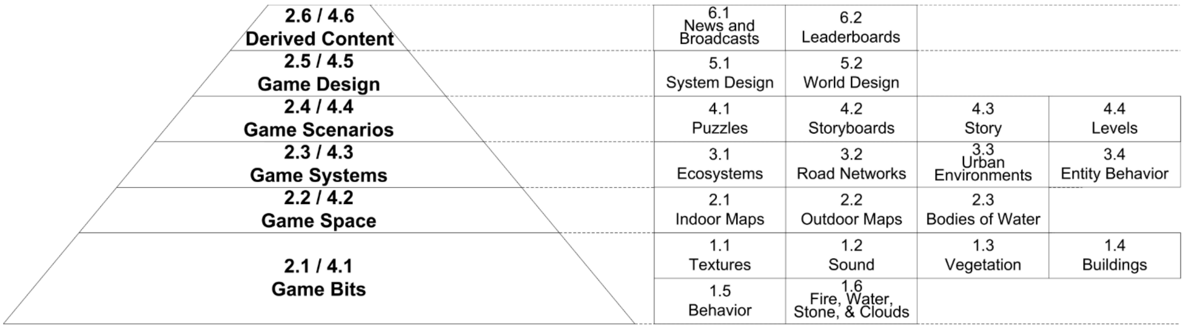


Figure 2.2: Six main classes of game content, as described and categorised by Hendrikx et al. (2013)

generation in games are alluded to, but much of the discussion is spent on procedural sound effects instead. Freiknecht and Effelsberg (2017) completely dismiss sound and music from their survey paper. This implies that the community has not yet begun to seriously tackle the application of generative algorithms to music in a video game. As can be seen in Fig. 2.2, Hendrikx et al. (2013) present six main classes of game content that may be generated using PCG. The authors state that these categories are presented in the form of a pyramid because elements towards the top of the pyramid may be created from elements from the bottom of the pyramid. According to this diagram, *sound* (which also incorporates music) falls under the *game bits* category, and should therefore be an essential component in the generation of content found in higher layers of the pyramid.

A taxonomy for PCG algorithms for games is provided by Togelius et al. (2011b), and is as follows:

online* versus *offline

online algorithms run during the game's runtime, while offline algorithms are run during the game's development and the results finalised during release.

necessary* versus *optional

necessary content is required by players to progress through the game, while optional content may be ignored with no detrimental value to completing a game. Togelius et al. (2011b) use this distinction to refer to content that affects gameplay (such as levels, bosses, or in-game rules, all of which may be considered optional depending on the game in question) as per their definition of game content described earlier. Under this distinction, music would be normally classified as optional content (since outside of rhythm games, music is not normally considered to be essential to completing a game).

random seeds* versus *parameter vectors

also referred to as the *degrees of control*, or the amount of parameters the algorithm takes in order to generate the result. While a seed consists of only one value (and therefore gives the algorithm a low level of control), having multiple parameter vectors allows the algorithm to be modified in multiple different ways by the user.

stochastic* versus *deterministic

the amount of variance in the results that the algorithm produces. If an algorithm produces the same results when the same parameters are given, then it is deterministic.[19]

constructive* versus *generate-and-test

constructive algorithms generate the content once, making sure that they never generate broken content, while generate-and-test algorithms keep generating results and testing them until they find something suitable

Following this taxonomy, a generative music algorithm that focuses on increasing immersion while taking into account player interactivity would have to be *online*, since it would be running at the same time as the game is being played, and *optional* since it is not generating content that is essential to the gameplay. It would have to have a high

[19]Note that deterministic algorithms can also be interpreted as a form of data compression; *.kkrieger* (.theprodukkt 2004) is a good example of an FPS that uses procedural content generation to compress the whole game into 96kb.

degree of control to be able to interpret the game state and generate music accordingly, and should be *stochastic* to allow for different music each time. Finally, it should be *constructive*, since it may not have the opportunity to generate and test the results during gameplay (though if done quickly enough, this could certainly be a viable approach).

2.2.4 Algorithmic Music in Games

Jørgensen (2008a, p. 163) describes what she calls the *golden rule of audio design* – "never let the player become annoyed or bored by the repetitiveness of the sound". This is a core tenet of my work and a driving force behind research in generative music for games, and following this golden rule is critical to my focus on generating immersive music. Algorithmic techniques are frequently used in games to allow the music to change and react to changes in the game or its environment, thereby following this golden rule of audio design. There have been several algorithmic approaches to music in computer games, both in academia and in industry, and this section discusses several of these techniques, as well as the ways they have been used.

Medina-Gray (2016) describes the concept of modular music used in avant-garde music, discussing concepts such as open form and aleatoric music that have since been used algorithmically in games. For example, Medina-Gray makes reference to Stockhausen's *Klavierstück XI* (Stockhausen 1956) where the pianist must choose which segments to play in which order, and compares this process to the computer randomising[20] pieces of the score in the action-adventure game *The Legend of Zelda: The Ocarina of Time* (Nintendo EAD 1998). Similar ideas are described by Aristopoulos (2017) (the composer for the action platformer game *Apotheon* (Alientrap 2015)), who designs an algorithmic system focused on recombinant techniques to improve the linear looping of music in games.

[20]Medina-Gray (2016, p. 61) takes a closer look at the role of both the computer and the player, stating that "[w]hile chance enters modular assembly through the computer, choice in the assembly step is the purview of the player." A thorough examination of this concept is considered to be out of scope for this dissertation.

One of the six techniques described by Langston (1989) is the "riffology" algorithm, which was used in the sports game *Ballblazer* (Lucasfilm Games 1984). Langston states that this algorithm was inspired by his experience as a lead guitarist. Essentially, the algorithm has a record of melodic fragments (or riffs) that can be used by the algorithm. When one is selected at random, the algorithm determines at what volume the melodic fragment should be played at, at what speed it should be played at, and whether or not to add or remove notes from the fragment, among other options. The riffology algorithm is restricted to notes from a blues A scale. However, as stated by Langston (1986, p. 5), while the results of the riffology algorithm may be considered music, they cannot really be considered interesting music, "because the rhythmic structure and the large scale melodic structure are boring. It appears that for music to remain interesting it must have appropriate structure on many levels."

For the action-adventure/survival game *No Man's Sky* (Hello Games 2016), Weir (2017) states that the musical system *Pulse* collects music stems written by the experimental/post-rock band 65daysofstatic and automatically organises and layers them into suitable soundscapes based on environment and what the player is doing in-game. The battle royale game *Pixelfield* (Epic Stars Inc. 2016) is claimed by Barreau (2018) to be the first game to feature an entire soundtrack that was generated by an algorithm. Here, a melody is created offline by AIVA using deep learning networks that are trained to generate music in a particular style, and the results tweaked and orchestrated in a digital audio workstation later (Frey 2019). This implies that the algorithm cannot yet function in a real time environment.

The simulation/strategy game *Spore* (Maxis 2008), which allowed players to create their own creatures, buildings, vehicles, and spaceships, implemented several constraints in their music generator using PureData according to Jolly and McLeran (2008). For example, notes are chosen from one specific scale, and seeds used to create short rhythmic and melodic pieces that can be looped and modified. Furthermore, sample choices are restricted depending on the type of object being built in the game. The top-down beat 'em up game *Ape Out* (Cuzzillo 2019) restricts the scope of algorithmic composition by only focusing on a particular genre of music; in this case, by generating a percussive jazz soundtrack by simulating a drummer.[21]

There are several recent examples of generative music systems for games that have appeared in academic literature. Hoeberechts et al. (2014) describe one such system called the *Algorithmic Music Evolution Engine*, which is a component based system architecture that generates music in real time based on game events. Different generators are used to create various parts of the musical piece, such as generators for the musical structure, harmonic content, motifs, and meter, and may either generate material randomly or make use of patterns. Finally, producer components use these different generators in order to generate music in a structured fashion. Hoeberechts et al.'s system was implemented in the unpublished match-3 game *Pop Tunes* (Hoeberechts et al. 2012).

Gungormusler et al.'s (2015) work is based on work done by Hoeberechts et al. (2014), but contains an architecture that is similar in structure to a musical performance by using musician components that are directed by a conductor component. The authors state that different generators are used to control the structure of the music at various different levels, and are also able to choose between synthesising sounds or using samples. Gungormusler et al. claim that when testing the system with users, the vast majority of people were able to determine that the music was changing in real time.

Several techniques for algorithmic composition in games focus on dynamic music that can convey different emotional states being portrayed in a game. For example, Prechtl et al. (2014a) describe the use of an algorithm that makes use of Markov models that is

[21]A demonstration of the algorithmic music system used in *Ape Out* can be seen at `https://www.youtube.com/watch?v=ed0wicQ6rFs`.

able to move between music that portrays safety and music that portrays danger (such as by decreasing the probability of major chords and increasing the tempo; this was successfully implemented in the maze game *Escape Point* (Prechtl 2015)). The *Sonancia* system by Lopes et al. (2015) is able to generate music with varying levels of tension based on a procedurally generated level built around a specified tension curve. Here, the music is created using various short pieces that can be dynamically combined depending on the level of tension required. Scirea et al. (2018) use evolutionary algorithms that generate melodies that include wanted features while avoiding unwanted ones; this was implemented in an open-source version of draughts.[22]

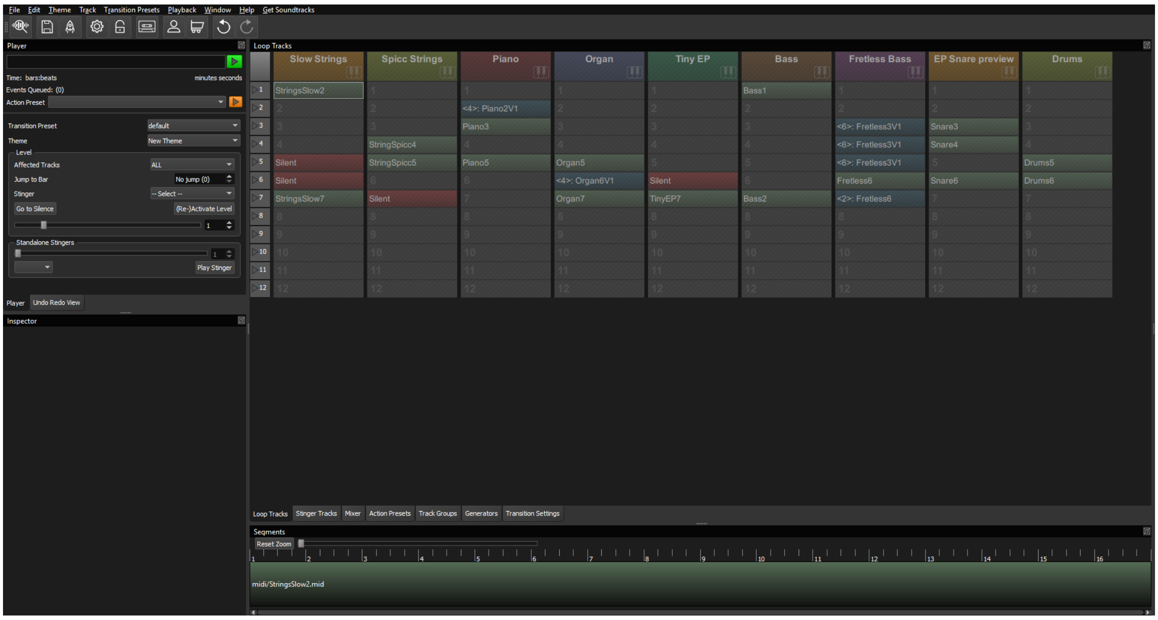


Figure 2.3: A screenshot of the adaptive music software *Elias Studio* (Elias Software 2015)

Various third-party tools exist that allow composers and audio programmers to harness algorithmic techniques without having to implement everything from scratch. One such example is *Elias Studio* (Elias Software 2015), illustrated in Fig. 2.3, which allows composers to use a multitude of techniques in order to ensure that the resulting music sounds acceptable to the composer.

[22]Also known as checkers in American English.

As has been demonstrated above, algorithmic music in games has been implemented using a variety of techniques and to solve a variety of different problems. The rest of this dissertation will focus on using algorithmic techniques in order to generate transitions between two pieces of music in a real time environment such as a video game. As will be discussed in Section 2.3, there are still considerable gaps in this area and in its understanding, making it a suitable avenue for further research.

2.3 Transitions

This section discusses transitions as a musical phenomenon that occurs across different levels of musical form, as well as how transitions have been composed historically and implemented in interactive media.

2.3.1 Definition

The term *transition* has been used to refer to a variety of different concepts, and several other terms have be used to refer to similar ideas. This section will attempt to sort through these different definitions in order to isolate the definition that is most useful in this dissertation.

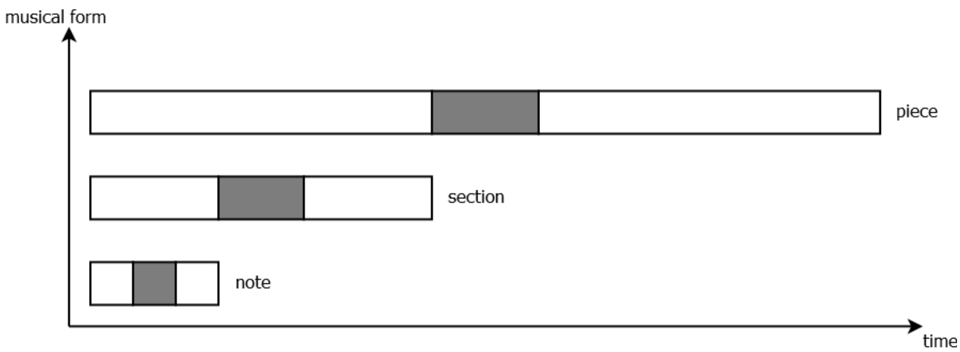
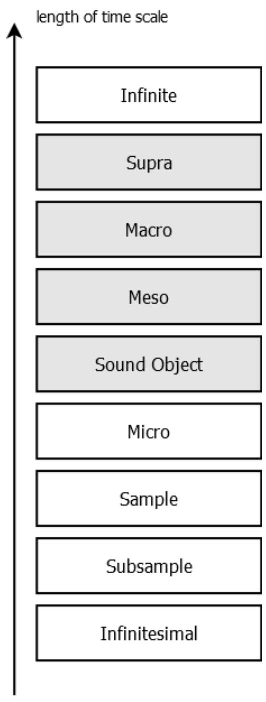


Figure 2.4: Hierarchy of form for the different definitions of *transition*

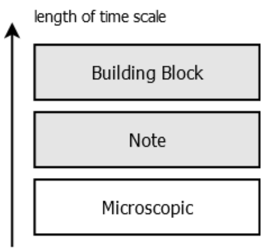
Fig. 2.4 illustrates the placement of the different uses of the term *transition* in the literature in terms of their hierarchical placement in musical form, where the transition is shown using the colour grey.

As can be seen in the diagram, three different types of transitions have been highlighted:

- **inter-piece transitions**, which handle transitions between two pieces of music
- **inter-section transitions**, which cater for transitions between two sections in a piece of music
- **inter-note transitions**, which encompass transitions between two notes in a music section



(a) Roads's nine time scales of music (Roads 2001, pp. 3–4)



(b) Miranda's three musical levels of abstraction (Miranda 2001, pp. 2–3)

Figure 2.5: Comparison between musical time scales

This hierarchical placement mirrors the nine time scales of music defined in Roads (2001, pp. 3–4), as well as Miranda's (2001, pp. 2–3) three musical levels of abstraction. These can be seen as illustrated in Fig. 2.5.

Due to the differences in size between each time scale and musical level described above, this dissertation will be focusing on a subset. Four relevant time scales taken from Roads (2001, pp. 3–4) are briefly described below, while two musical levels of abstraction taken from Miranda (2001, pp. 2–3) have been appropriately grouped together with Roads's definitions.

supra

Roads describes this musical time scale as being longer than any single composed piece of music, and extends into "months, years, decades, and centuries".

macro

This time scale is described as spanning a single composed piece of music, normally consisting of minutes, hours, or in extreme cases, days.

meso

As stated by Roads, this time scale handles "divisions of form" and "groupings of sound objects", such as musical phrases. This time span is usually measured in seconds or minutes. Miranda refers to this level of musical abstraction as the *building block* level.

sound object

This time span is described as being "a basic unit of musical structure", and normally covers single notes or events on that scale. It is normally measured between fractions of a second to a few seconds. This level is referred to by Miranda as being the *note* level.

Based on the time scales given above by Roads and Miranda, the hierarchy of form for transitions can be mapped as followed:

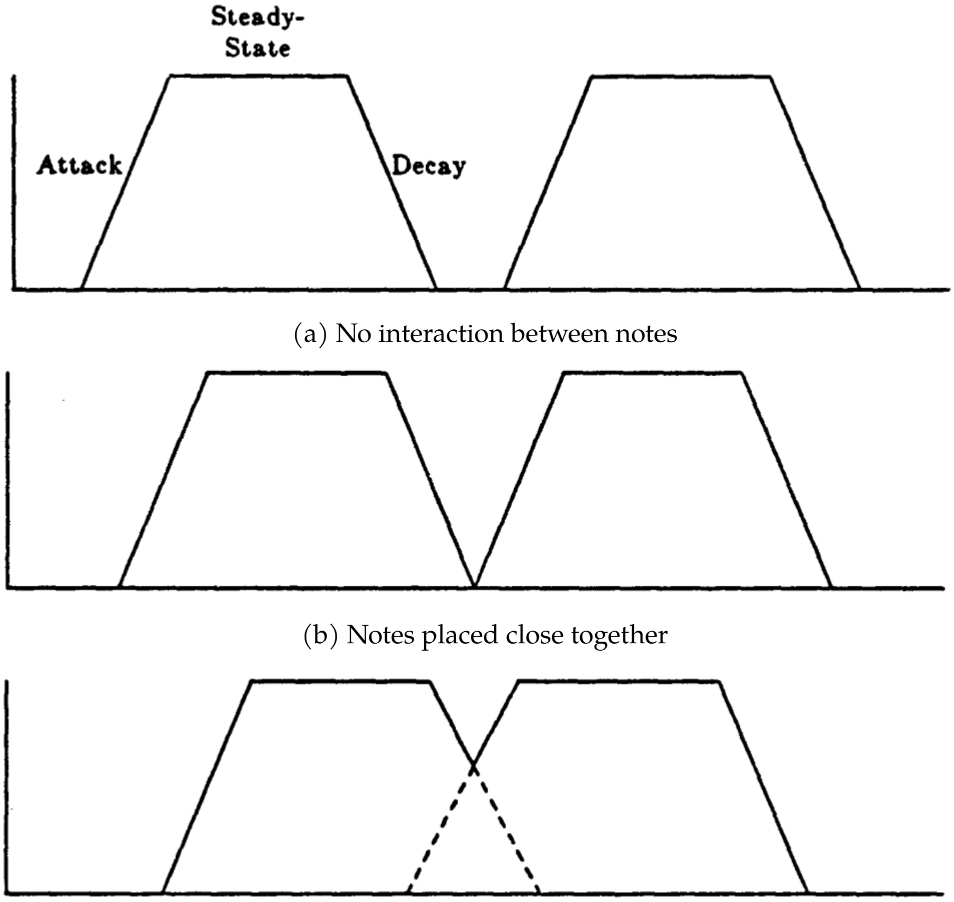
- **inter-note transitions** operate on a *sound object* scale to create music on a *meso* scale
- **inter-section transitions** operate on a *meso* scale to create music on a *macro* scale
- **inter-piece transitions** operate on a *macro* scale to create music on a *supra* scale

I will give details regarding each individual transition layer in the following sections.

2.3.1.1 Inter-Note Transitions

Inter-note transitions are transitions between individual notes in a piece of music. These transitions normally deal with parameters for audio waveforms rather than with a symbolic representation like the commonly used MIDI representation.

According to Strawm (1985), a transition is the overlap between the attack and decay of two successive notes, and "includes the ending part of the decay of one note, the beginning and possibly all of the attack of the next note, and whatever connects the two notes" (Strawm 1985, pp. 2, 175); this is illustrated in Fig. 2.6. These types of transitions take up a small amount of time, typically in the tens or hundreds of milliseconds.



(a) No interaction between notes

(b) Notes placed close together

(c) Notes overlapping

Figure 2.6: Inter-note transitions (reproduced from Strawm (1985, p. 2))

While Strawm (1985) uses discrete short-time Fourier Transform as a technique to generate suitable waveforms as transitions between two notes, Holloway and Haken (1992) use an extended McAulay-Quatieri technique[23], which they claim also allows them to generate transitions between non-harmonic sounds. Both Strawm and Holloway and Haken state that the best way to approach the synthesis of inter-note transitions is to

[23]The *discrete short-time Fourier Transform* and the *extended McAulay-Quatieri technique* are both algorithms used in digital signal processing. I will not look in detail at these algorithms, because the focus of this dissertation is primarily on the generation of musical transitions using symbolic representations rather than music waveforms.

overlap the two notes slightly and synthesise the overlap to successfully bridge the two notes.

I will not be considering inter-note transitions in this dissertation for several reasons. First of all, these techniques can only be applied to audio waveforms, while this project attempts to model transitions between different pieces using a symbolic representation of music. This means that audio synthesis is never involved. Secondly, with reference to Fig. 2.5, the level of musical form at which inter-note transitions take place is too small to be considered useful when generating transitions between pieces of music in a video game, and inter-section transitions or inter-piece transitions would be more appropriate.

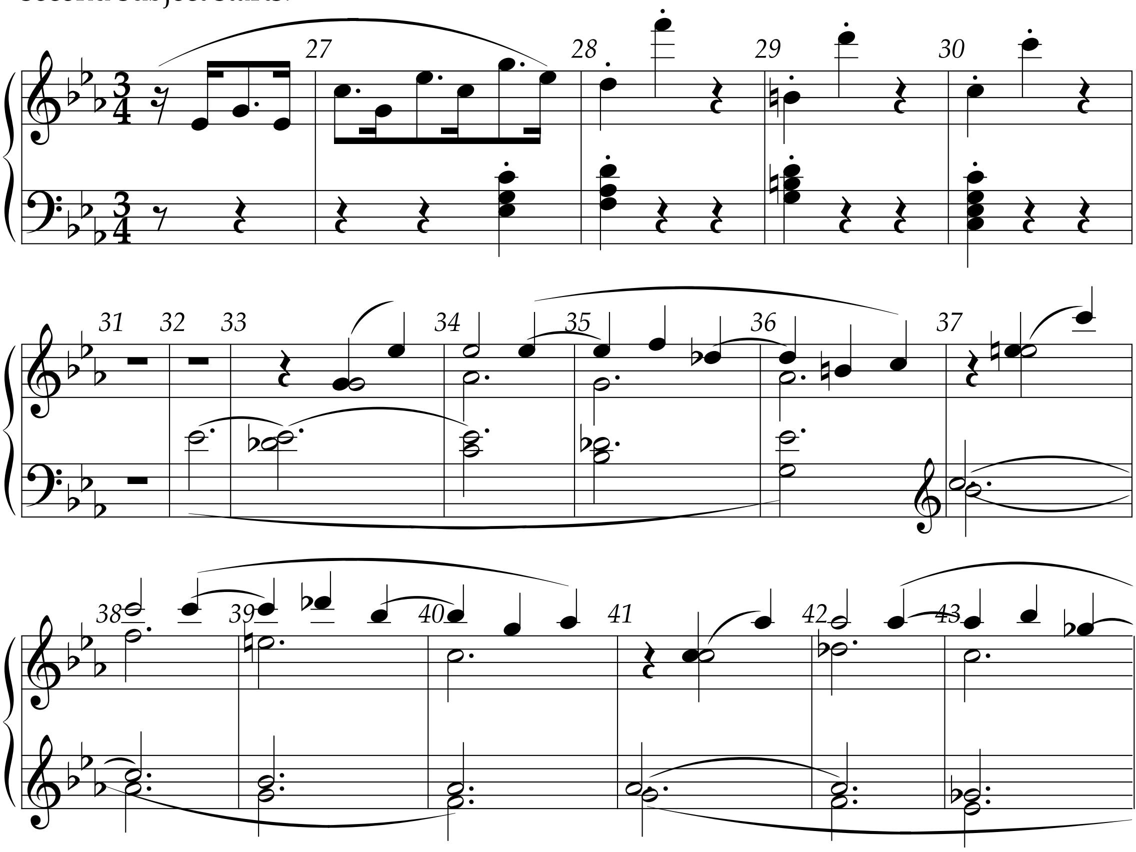
2.3.1.2 Inter-Section Transitions

Inter-section transitions are transitions between different sections in the same piece of music. One example of this is described in Macpherson (1930) when referring to the sonata form, a tripartite form built up of three parts: *exposition* (notated as A), *development* (notated as B, and serves as a contrast to A), and *recapitulation* (notated as A′, and serving as a reintroduction of the ideas expressed in A). The sonata form is discussed here since it can be used as a point of reference as to how musical transitions have been handled within the composition of classical music, and how the techniques used there may be used when generating musical transitions.

Here, Macpherson states that the transition is “matter serving as a connecting-link between [the first subject] and the Second subject” (Macpherson 1930, p. 122) within the exposition. These transitions may either use elements from the first section or introduce entirely new musical material (Macpherson 1930, p. 122). [24]

[24] Interestingly, Macpherson (1930, pp. 122–123) states that a transition was considered a novelty towards the end of the 18th century. It was first introduced using “scale-passages and broken-chord figures” in order to introduce this new type of structure to listeners and to serve as sign posts, informing them that something new was being introduced. Once this was established, composers starting making these transition pieces more creative.

Macpherson, pp. 123–125 states that *Piano Sonata No.5, Op.10 No.1* by Beethoven is an example of a piece where the transitions introduce new musical material, as can be seen in Example 2.1 below.[25] According to Macpherson, the first subject comes to an end at bar 30, with the transition beginning at bar 32 and continuing till bar 56, where the second subject starts.



[25]Note that this musical example and all other musical examples presented in this dissertation are representations of the particular idea or theme being discussed, and do not necessarily conform to the normal rules of music presentation. For example, accidentals are repeated in a bar, and no end barlines are used.

Example 2.1: An example of a transition introducing new musical material from Beethoven's *Piano Sonata No.5, Op.10 No.1*

Similarly, *Piano Sonata No.14, Op.27 No.2* by Beethoven (1975a) is given as an example where the transition uses material from the first section. Fig. 2.2 shows an example from the third movement, where according to Macpherson, the transition begins at bar 15 and ends at the crotchet in bar 21.

Example 2.2: An example of a transition as an outcome of the first subject from Beethoven's *Piano Sonata No.14, Op.27 No.2*, third movement

Schoenberg states that "[t]he purpose of a transition is not only to introduce a contrast; it is, itself, a contrast" (Schoenberg 1970, p. 178). Similarly, White (1984, p. 59) states the term *transition* should be used when referring to musical passages that modulate between sections and where the musical material in the first section must move to the contrasting key. However, this only applies to large forms such as the sonata-allegro and rondo forms.

Schoenberg (1970, p. 179) states that a transition is composed of four elements, as detailed below:

establishment of the transitional idea

Here, Schoenberg states that the transitional idea can be introduced through repetition, often in a sequential manner.

modulation

This musical feature is the source of contrast in the transition. According to Schoenberg, "a well-developed modulatory passage between the principal and subordinate themes becomes a valuable source of contrast" (Schoenberg 1970, p. 203).

liquidation of motival characteristics

Liquidation is described as a process concerned with "gradually eliminating characteristic features, until only uncharacteristic ones remain, which no longer demand a continuation" (Schoenberg 1970, p. 58).

establishment of a suitable upbeat chord

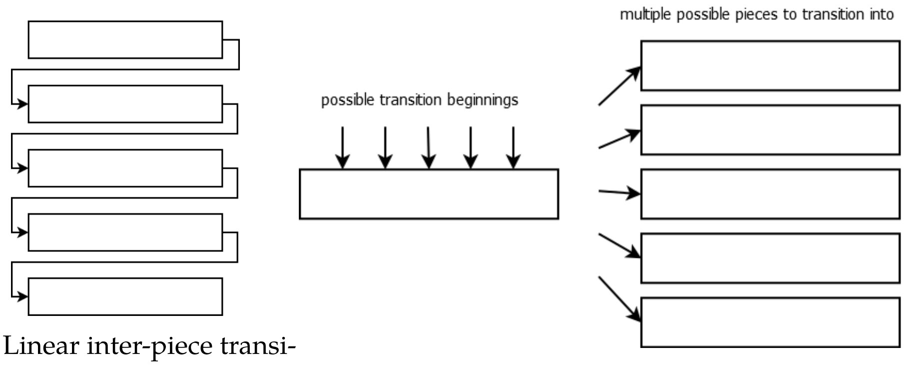
Establishing an upbeat chord allows the new section to be developed.

2.3.1.3 Inter-Piece Transitions

Inter-piece transitions are transitions that connect two separate and distinct pieces of music, or as defined by Sweet: "the musical bridge that connects the two pieces of music" (Sweet 2014, p. 166).

Unlike the previous two types of transitions, the point at which an inter-piece transition takes place in a piece of music must depend on an external event taking place, such as in a video game, rather than the usual unfolding of the piece.

Specifically, Sweet describes these pieces as being "source music" and "destination music"; in the context of a video game, these refer to a piece of music that the player is coming from, and a piece of music that the player is heading towards. For the rest of the dissertation, the terms *source piece* and *target piece* will be used.



(a) Linear inter-piece transitions

(b) Interactive inter-piece transitions

Figure 2.7: Differences in inter-piece transitions between linear media and interactive media

When writing inter-piece transitions, it is therefore important to know in what order the pieces will be played. This allows the music to be written in such a way as to follow seamlessly from one piece to another. This is particularly relevant for album releases[26], music playlists, and DJ mixes, which are all designed to be played in a linear manner from beginning to end. However, this is more complicated in interactive media such as video games, since there may be more than one possible piece that could follow the currently playing piece. In essence, linear media such as playlists, mashups, and mixes

[26] A good example of this is the album *Nonagon Infinity* by Australian psychedelic rock band King Gizzard & the Lizard Wizard (King Gizzard & the Lizard Wizard 2016), which features 9 tracks designed to seamlessly transition from one to the other, as well as have the entire album seamlessly loop, essentially creating an infinite track.

work in a similar manner to lists, while interactive media (like video games) works more like a graph; this difference is illustrated below in Fig. 2.7.

The following sections delve into more detail on how inter-piece transitions are used in their respective contexts.

2.3.1.3.1 Playlists, Mashups, and Mixes

While this dissertation will not examine automatic playlist generation in detail, a technique is presented below that has been used to generate suitable linear inter-piece transitions.

Davies et al. (2014) describe *AutoMashUpper*, a system that allows the automatic creation of mashups based on a selection of songs. The authors describe a *mashability* measure which is calculated based on harmonic compatibility, rhythmic compatibility, and spectral balance. This is calculated locally over various segments, as opposed to having a global score for the entire song. However, there are several disadvantages with using this technique in a real time environment. For example, songs are mashed together based on segments and will therefore still be subjected to a slight delay until the next possible position.[27] Furthermore, *AutoMashUpper* works with waveforms as opposed to a higher order representation of music, like MIDI. This means that several algorithms must be run in order to extract information such as harmonic capability and tempo from the waveform.

2.3.1.3.2 Looping

One sub-problem for inter-piece transitions is how to handle the looping of the same piece over and over again, meaning that the source piece and target piece are effectively the same piece. This is quite common in a video game, for example: when the length of

[27] This is similar to the way horizontal resequencing transitions work, as described in Section 2.3.3.4.

a specific piece of music is shorter than the amount of time a player spends in a location in a game. Looping is very often used in games since composers cannot know how much time a player will spend in a particular area. Early game composers also looped the music they composed to save space. However, K. Collins (2007, p. 278) claims that some players experience *listener fatigue* due to looping, which she defines as players becoming tired of listening to the same piece of music being repeated over and over. While I will not be taking a detailed look at techniques to improve looping, some relevant techniques that could be used when working with musical transitions are briefly discussed below.

Sweet (2014, pp. 128–130) and Phillips (2014, p. 168) both provide several techniques for composing effective loops and solving what Phillips calls the *loop point problem*: how to create a seamless transition between the ending of the piece and the beginning of the piece. Several of these techniques revolve around the concept of composing around the *consequence-antecedent*, where the end of the piece is composed to naturally resolve at the beginning of the piece when looped. This naturally means that instead of approaching the composition of a piece as having a beginning, a middle, and an end, the composition should be structured less rigidly; Sweet suggests breaking down the entire piece into smaller chunks that can be looped accordingly. Both authors do discuss the idea of avoiding *landmarks*: sonic features that stick out when the piece is looping and that can therefore break immersion.

2.3.1.3.3 Interactive Transitions

The previous examples that have been discussed have all used compositional techniques while creating the music in order to transition between one piece and another. Due to player interaction in video games, composers can never really predict when this change is going to take place, and the transition must therefore take place in real time, as can be seen in Fig. 2.7b.

2.3.2 Morphing

Methods such as morphing may be adapted to create suitable transitions. Typically applied to video, Slaney et al. (1996, p. 1001) define morphing as "a process of generating a range of images that smoothly move from one image to another". Similarly, audio morphing becomes a process of generating a range of notes to move from one melody to another, defined by Polansky (1992, p. 57) as shown in Equation 2.1:

$$f(S, T, \Omega) = M \tag{2.1}$$

where S is the source music, T is the target music, M is the resulting morph music, and Ω is what Wooller (2007, p. 4) describes as the morph index, such that when $\Omega = 0$, $M = S$ and when $\Omega = 1$, $M = T$.

Wooller and A. R. Brown (2005) identify three different categories of morphing: *compositional morphing*, *parameter morphing*, and *morphing in music theory*. These are described in more detail below:

2.3.2.1 Compositional Morphing

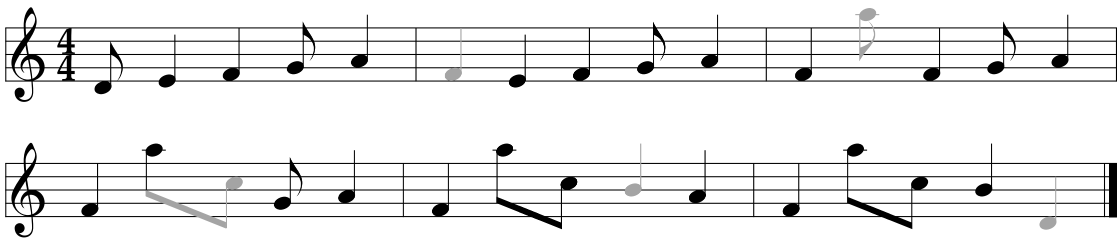
Compositional morphing is a "process that can generate a sui generis musical work by reworking musical elements extracted from any number of music inputs" (Oppenheim 1997, section 4). Wooller and A. R. Brown state that the main focus of compositional morphing is that of creating suitable morphs by using pitch and rhythm.

Some previous work has been conducted in this area, such as described in an abstract by Berger (1995), which unfortunately has not attracted further work, as well as commercial applications such as Edlund's (1999) *Musifier Adaptive Improviser* and Oppenheim's (1997) DMorph system. Both systems work in a similar manner where music is generated based on different weightings of music pieces, and can function both offline as well as in a real time environment.

Several discussed techniques can be categorised as *compositional morphing*, as described below:

cyclostationary

Slaney et al. (1996, p. 1002) describe *cyclostationary* morphing as "smoothly changing a repetitive sequence of sounds" using a sequence of steps. An example of this technique can be seen in Example 2.3



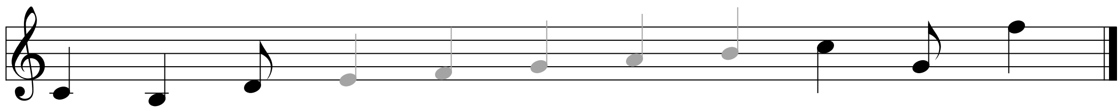
Example 2.3: The resulting piece using the cyclostationary transition algorithm. New notes that have been introduced are marked in grey.

interleaving

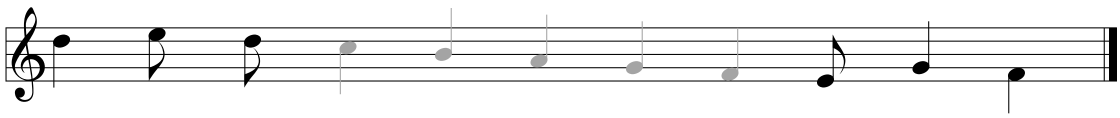
Interleaving is defined by Oppenheim (1997, section 6) as "alternately selecting one or more elements from each of the music inputs", such as sequences of pitches. A related concept to interleaving is what Wooller and A. R. Brown (2005) defines as *layering*, which introduces a weighted selection that determines which piece to select elements from.

interpolation

Interpolation is described by Wooller and A. R. Brown (2005) as being "a technique for estimating appropriate transition values between known points", with several different techniques available. One such example, linear interpolation, can be seen below in Examples 2.4 and 2.5, where the technique is used to transition between two different pieces of music.



Example 2.4: A rising linear interpolation



Example 2.5: A falling linear interpolation

weighted averaging

Wooller and A. R. Brown (2005) describe a technique they call *Markov morphing*, where two Markov chains[28] are created. In order to generate a transition, a model is chosen via weighted selection (where models are weighted depending on the current position in the transition) and a suitable musical element is generated.[29]

2.3.2.2 Parameter Morphing

Wooller and A. R. Brown state that parameter morphing involves the manipulation of sonic properties of music such as timbre, which according to the authors may consist of "interpolating filter coefficients, FM modulation ratios and depths, or other synthesis parameters".

[28] Markov chains are discussed within the context of algorithmic composition in Section 2.2, and described in further detail in Chapter 5.

[29] This transition technique is evaluated in Chapter 3 alongside other transition techniques used in games, and serves as inspiration for the novel algorithm presented in this dissertation in Chapter 5.

Since this dissertation deals with music represented in MIDI, and is focused on generating melodic transitions, techniques grouped as *parameter morphing* by Wooller and A. R. Brown are considered to be out of scope.

2.3.2.3 Morphing in Music Theory

Wooller and A. R. Brown state that morphing is an "established compositional strategy", with Oppenheim (1995) stating that Beethoven was perhaps one of the first composers to consciously attempt musical morphing between two distinct themes in his sonatas.[30] Modulation is also discussed as being a form of music morphing. Oppenheim (1997) also states that similar morphing techniques have been used outside of classical music, such as in electronic music and musique concréte "for modulating one sonic-texture into another".

2.3.3 Transitions in Games

So far, the types of transitions described in Section 2.3.1 have been discussed within the context of a non-interactive environment. This section describes the use of different transition techniques in interactive environments such as video games.[31]

As stated in Section 2.1.3.1, music has different uses in games, which Sweet (2014, pp. 329–330) groups into three categories: closed systems, open systems, and combat-triggered music interactivity. These categories similarly reflect the different use cases of musical transitions in games, as follows:

zone transitions

This category of transitions deals with music changes when moving between

[30] This is discussed in more detail in Section 2.3.1.2, in the context of *inter-piece transitions*

[31] As discussed in Section 2.1.3.3, video games tend to use different musical genres and tempos in order to accommodate for different in-game scenarios. One notable exception is the Western-themed action/ adventure game *Red Dead Redemption* (Rockstar San Diego 2010), where composers restricted themselves to writing pieces in A minor using 130 or 65 beats per minute, in order to make it easier to transition between different pieces (Jeriaska 2011).

two different in-game locations. These locations often have unique music associated to them. Zone transitions may be seen in open-world games such as *World of Warcraft* (Blizzard Entertainment 2004), where different areas are associated with their own pieces of music, as well as closed-system games such as in the point-and-click adventure game *Monkey Island 2: LeChuck's Revenge* (LucasArts 1991) or the JRPG *Final Fantasy X* (Square 2001).

event-triggered transitions

This category contains any possible transition to a piece of music that is triggered by some sort of in-game event. One common example is a *combat-triggered transition*. In this case, musical transitions are used to move between a *resting state*, which normally contains music that the player experiences when exploring the world, and *combat state*, containing music composed specifically to signify that the player is now in combat against an enemy. This applies to games where combat is considered a separate game state (such as *Final Fantasy X* (Square 2001) or *Pokémon Red* (Game Freak 1996)) as well as games where the player seamlessly enters combat (such as in the open-world action game *Batman: Arkham Knight* (Rocksteady Studios 2015)).

Other transitions that could fall under this category include *narrative transitions*, where the music changes due to some narrative event in the game (such as triggering a cutscene) and *status transitions*, where the music changes to reflect a change in the player character's state. The most common example of this is when the character is dangerously close to dying, such as in *Pokémon Red* (Game Freak 1996) where alarm bells are added to the melody.

Five transition techniques are frequently described in the literature; these techniques are the *abrupt cut transition*, *crossfading*, *stingers*, *horizontal resequencing*, and *vertical re-*

orchestration, and are discussed in further detail below. Other novel transition techniques will be discussed towards the end of this section.

2.3.3.1 Abrupt Cut

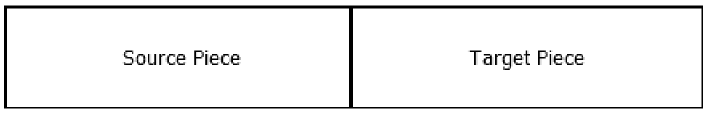


Figure 2.8: Abrupt cut transition

Early games tended to simply cut abruptly from one song to another when needed, as can be seen in Fig. 2.8.[32]

One example of the use of this transition is in the arcade game *Cavelon* (Jetsoft 1983). Here, players control a knight that storms a castle in order to rescue a princess. Fast and upbeat music is played as players explore each level of the castle; however, if the player character is shot by one of the guards inside the castle, an abrupt cut transition is heard as the gameplay music stops playing and death music is played instead. Another example is in the arcade game *Frogger* (Konami 1981), where players must guide a frog home by crossing a busy road and a dangerous river. Once players managed to successfully guide a frog home, the music would abruptly change to a new piece.

While implementing an abrupt cut transition is as simple as stopping the currently playing piece of music and starting the new piece, the downside of using an abrupt cut transition is that if the source and target pieces are different enough from each other, they will audibly clash.

[32] An abrupt cut transition is also referred to as *butt-splicing* within the audio engineering community; see Eargle (2003, pp. 338–341).

2.3.3.2 Crossfading

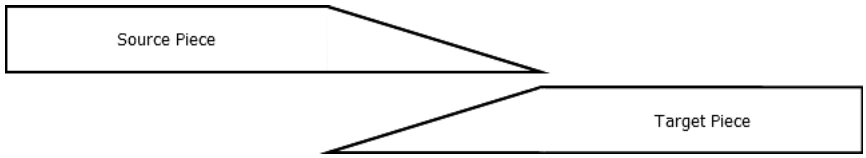


Figure 2.9: Crossfade transition

In order to transition between different pieces of music without resorting to an abrupt cut transition, one possible solution is to crossfade between the tracks when needed. This is done by lowering the volume of the initial piece while gradually increasing the volume of the next piece to be played. An illustration of this type of musical transition can be seen in Fig. 2.9, and as shown in the figure, it may be possible to hear both pieces of music playing at the same time as the volume is being adjusted.

A crossfade typically occurs between two pieces of music, but may also occur between a piece of music and silence; here, the music is described as *fading out* to silence, or *fading in* from silence. In both scenarios, the only parameter that can be changed is the speed of the crossfade.

Crossfading is overwhelmingly one of the more popular techniques used in video game music, since it is a relatively simple technique to implement. However, there are scenarios where a crossfade may result in a jarring transition, such as when the pieces are of wildly different musical genres (such as rock, jazz, or trance), moods, tempos, or meters.[33] An example of such a jarring transition can be found in *World of Warcraft* (Blizzard Entertainment 2004) when travelling between the Asian-inspired region called the *Valley of the Four Winds* and the horror-themed region called the *Dread Wastes*, and partic-

[33]Note that if the meters align, crossfading between two pieces of music may be slightly easier. A more extreme example can be seen in *Red Dead Redemption*, where as previously stated, all pieces in the game were composed at a tempo of 130 beats per minute or the corresponding half time of 65 beats per minute to make for easier transitions (Jeriaska 2011).

ularly between the tracks mus_50_bamboo_01 and mus_54_shadescendant_01.[34] A crossfade between these tracks is jarring due to extreme difference in tempo and genre between the two pieces.

2.3.3.3 Stingers

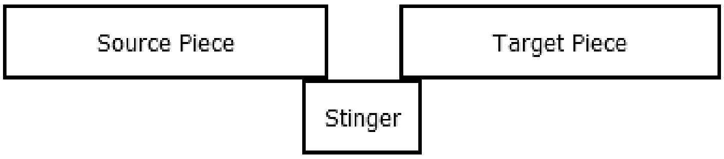


Figure 2.10: Stinger transition

In an attempt to hide the change in music, some games have used a *stinger* – a short piece of music or a sound that masks the change between one piece of music and another. This is illustrated in Fig. 2.10. The JRPG *Final Fantasy X* (Square 2001) is one example of this; as players travel to different areas, scene-appropriate music is played to fit the area, such as the piece ビサイド島 (*Besaid Island*)[35] when the player is exploring Besaid Island. However, when the player comes across a random encounter and has to battle monsters, the encounter is introduced with a visual effect (the shattering of the screen) and an appropriate sound cue before the battle music ノーマルバトル (*Battle Theme*)[36] begins playing. The sound of the screen shattering masks the transition between the area music and the battle music.

[34]The track mus_50_bamboo_01 can be heard at the following link: https://www.youtube.com/watch?v=Y-0gMlSg1gE, while the track mus_54_shadescendant_01 can be heard at the following link: https://www.youtube.com/watch?v=9ta1fCfL0tE&t=435s.

[35]The track ビサイド島 or *Besaid Island* can be heard at the following link: https://www.youtube.com/watch?v=cMYasM_o2hU&t=2755s

[36]The track ノーマルバトル or *Battle Theme* can be heard at the following link: https://www.youtube.com/watch?v=cMYasM_o2hU&t=1293s

2.3.3.4 Horizontal Resequencing

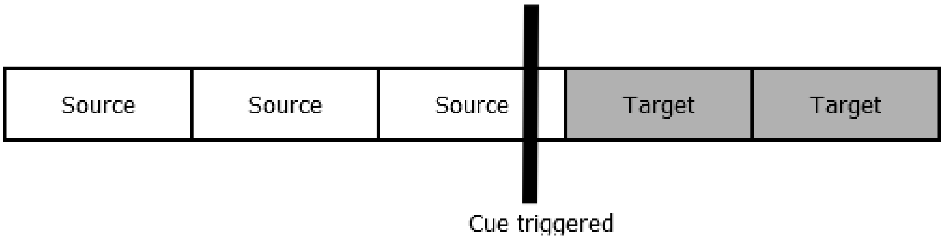


Figure 2.11: Horizontal resequencing transition

More complex approaches to transitioning music involve the use of horizontal resequencing or vertical reorchestration. *Horizontal resequencing*, also called cue-to-cue transitions, describes the scheduling of musical changes in advance depending on certain events that are called. An illustration of this transition can be seen in Fig. 2.11.

A well-known example of the use of horizontal resequencing in games is *Monkey Island 2: LeChuck's Revenge* (LucasArts 1991), a pirate themed point-and-click adventure game which makes use of the *iMuse* system (Land and McConnell 1994). A good example can be seen in Woodtick, a city in the game. As the player enters different buildings in the city, a cue is triggered that informs the audio engine to switch to more relevant music on a bar by bar basis (Strank 2013).

While horizontal resequencing allows the music to wait for the right moment in order to successfully transition between two pieces, one downside to this technique is the way that the triggered musical change will always lag behind the visual change, even if by a small amount. For example, as can be seen in Fig. 2.11, while the cue to change music has been triggered in the third bar, the game must wait for the third bar to finish playing before switching to the next piece of music. Paul (2013, p. 67) describes this time delay as a "speed quantisation", and introduces several possibilities, such as immediately[37], after

[37] An immediate speed quantisation means that an abrupt cut transition is used, since instead of waiting for the appropriate moment to trigger the transition, it simply occurs immediately.

each beat, after each bar, or after every phrase. Alternatively, custom markers may be added to each piece of music that indicate the best position for initiating a transition; the placement of these markers would of course vary depending on the type of music being used. As stated by Paul, the downside to custom markers is that they require manual placement.

2.3.3.5 Vertical Reorchestration

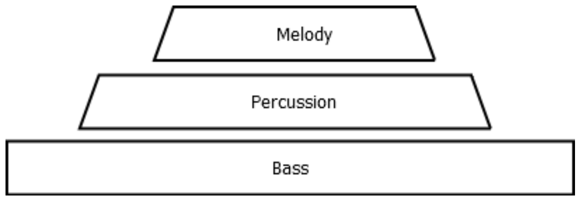


Figure 2.12: Vertical reorchestration transition

Vertical reorchestration involves the addition or removal of different musical elements or melody lines to the currently playing piece. This can be seen in Fig. 2.12.

One example of the use of vertical reorchestration is in the 3D action platformer *Banjo-Kazooie* (Rare 1998), where the technique is used to differentiate between being underwater or not. Similiarly, the strategy/puzzle game *Pikmin* (Nintendo EAD 2001) allows for seamless transitions by fading entire instrument lines in and out depending on the player's activity in the game, such as entering combat or ending the day.

2.3.3.6 Other Transition Techniques

One compositional technique that has been used to create successful transitions is musical fragmentation.[38] A clear example of its use can be seen in *The Legend of Zelda: Breath of the Wild* (Nintendo 2017). Using musical fragmentation allows the in-game music

[38]Musical fragmentation consists of splitting up melodies and musical motifs into smaller fragments. Sweet (2014, p. 117) states that many game composers were inspired by Stockhausen's *Klavierstück XI* (Stockhausen 1956), which makes frequent use of this compositional technique.

to be broken up into short musical sections and silence. Yu (2017) states that this provides various advantages, such as preventing players from anticipating the music (and therefore feeling tired of the music) as well as making transitions between different pieces of music easier.

Another useful concept for seamless transitions is imbricate audio. Hulme (2017) describes this as an "extension of the concept of a standard modular score", based on the process of "recording a score with regular pauses to capture the reverb tails of every branching point", as can be seen in Fig. 2.13. By breaking up music into modular chunks and preserving reverb tails, Hulme (2017) states that crossfading is no longer needed in order to transition between different pieces. However, this technique still suffers from the same problems that horizontal resequencing has; namely, that the triggered musical change will lag behind the music being played.

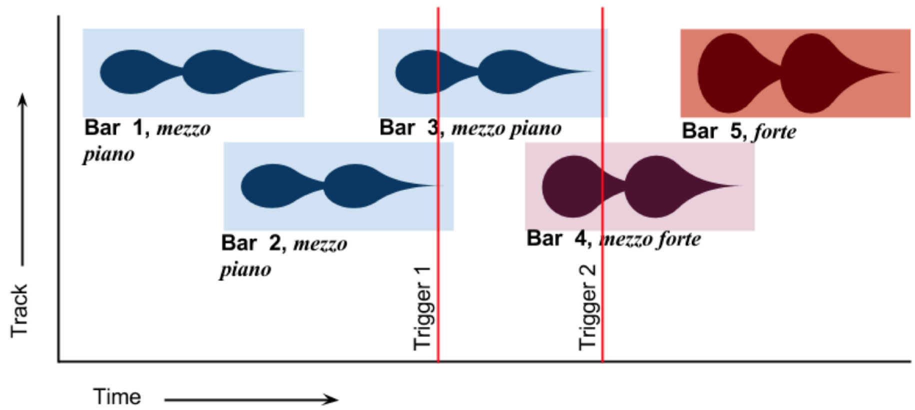


Figure 2.13: An example of imbricate audio in use between two different tracks (reproduced from Hulme (2017))

Some work has been also done on automatically generating musical transitions. One example is work done by Aspromallis and Gold (2016) on generating interactive transitions using musical grammars. Here, the authors cater for unpredictable interaction by generating chords based on three different events: `ENDING` (where by predicting when the player will reach the goal, a cadence is scheduled at an appropriate time during the currently playing music), `RECOVERED` (where the algorithm recovers and returns to the previously generated music if the player has changed their mind), and `GOAL_REACHED` (where the musical transition is complete). Two limitations to this system are as follows: first, there is no mention of any target music to be reached and transitioned into. Secondly, the algorithm as described by Aspromallis and Gold (2016) caters for chords rather than melody, limiting its use in a video game.

A similar approach is taken by Prechtl et al. (2014a) and used in the maze game *Escape Point* (Prechtl 2015); here, a first-order Markov model is used to generate chords as the player traverses through a maze. As players encounter enemies within the maze, the chords that are being generated change into ones that are more suited to depicting danger (such as by increasing volume and tempo, and preferring diminished chords over major chords). Here again, the generative system focuses on chords rather than melody. However, while the algorithm described by Aspromallis and Gold (2016) works with music transitioning due to changes in space (such as moving from one area to another), the algorithm described by Prechtl et al. (2014a) generates music based on changes in the game environment (such as the presence of enemies).

Gillespie and Bown (2017) use a geometric model of interval spaces in order to generate appropriate music for gameplay, relying on predeterminated events that determine when combat has started or ended to inform the system. However, the authors evaluate their system on a recording of gameplay as opposed to within the game environment itself, and the corresponding music is first generated and then synthesised and mixed in order to match the game recording's audio. While preliminary work, this approach certainly seems promising.

Sporka and Valta (2017) discuss the design and implementation of the Sequence Music Engine, a system used to generate seamless transitions in the action/RPG *Kingdom*

Come: Deliverance (Warhorse Studios 2018). Each pre-composed piece of music has its own start and ending (represented in Fig. 2.14 at the *own start* and *own end* positions). Seamless transitions are created by composing branches for each piece of music, allowing phrases to end gracefully and terminate at a "well-defined state of music which can be characterised by the current melodic tendency, harmonic function, and orchestration" which the authors call an aleph (represented with the symbol ℵ). Corresponding intro pieces can also start from the appropriate aleph in order to link up with the piece of music in question. This system is illustrated in Fig. 2.14. Sound effects called cinels are also used in the system and function in a similar manner to stingers, in that they mask any noticeable transition points between one piece and another as well as providing a suitable amount of suspense.

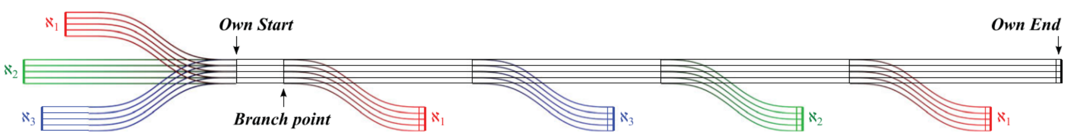


Figure 2.14: Seamless transitions using alephs and branching points (reproduced from Sporka and Valta (2017))

While the focus of this dissertation is on digital games, there are other domains that also make use of the aforementioned transition techniques. The board game *Dropmix* (Harmonix 2017) allows players to play the part of a DJ by adding cards to the board that represent different stems from well-known popular songs (such as drums, vocals, or guitars). This means that the gameplay revolves entirely around the concept of vertical reorchestration, with different stems being played or removed from the overall mix. Furthermore, the music also waits for the most appropriate time before adding a newly played stem (such as waiting from the appropriate beat or by introducing a drop[39]), demonstrating the use of the horizontal resequencing technique.

[39] A musical drop is an increase in tension and a sudden change in the rhythm.

Figure 2.15: *Syrinscape*'s user interface to allow for creating suitable soundscapes for TTRPGs

Another example of transitions used in TTRPGs is the use of *Syrinscape* (Loomes 2008), as seen in Fig. 2.15. Here, different layers of sound effects and small snippets of pre-recorded loopable music are collected together in the form of a soundboard and used to create a suitable atmosphere; in essence by making use of the vertical reorchestration technique. When transitioning between different scenes that would necessitate the use of different sound packs (such as changing from the Desert sound pack to the High Seas pack), crossfading is used so that any unnecessary sounds and music are slowly faded out and more relevant sounds and music are faded in.

Chamberlain et al. (2017) discuss what they define as *locative music*: music that is able to change and react depending on the physical location of a person in a defined area, as can be seen in Fig. 2.16. Here, pre-composed music is assigned to particular regions at The Yorkshire Sculpture Park, and as visitors walk in and out of these regions, a GPS-enabled phone can interpret where they are physically located and transition to an appropriate piece. The authors handle this approach by subdividing each region into bands that give the music the time needed to transition appropriately.

Figure 2.16: Locative music. Taken from Chamberlain et al. (2017, p. 29)

Note that the concept of locative music is similar to zone music as implemented in open-world games (as discussed in Section 2.3.3). An example of this can be seen in Fig. 2.17, which illustrates the different musical zones[40] present in the region of Darkshore, Kalimdor in the MMORPG *World of Warcraft* (Blizzard Entertainment 2004).

2.3.4 The Ideal Transition

Several criteria have been identified as being necessary in order to create the ideal transition. These have been grouped into two distinct categories: *rate of change* and *degree of fitting*, and will be discussed below.

[40]These musical zones, marked in various different colours, were added by the author and are not present in the original image.

Figure 2.17: A map of Darkshore, Kalimdor in *World of Warcraft*, with musical zones loosely mapped out (author's addition)

2.3.4.1 Rate of Change

Macpherson (1930, p. 128) states that the most important aspect of the transition is the "gradual obliteration of the key and feeling of the first theme"; indeed, he states that composing a transition is "often a test of the composer's constructive skill, as nothing reveals weakness in this respect more than what may be described as a bad "join"" (Macpherson 1930, p. 123).

Similarly, K. Collins states that abrupt cut transitions "can feel very jarring for the player" (K. Collins 2008, p. 146), and crossfading transitions, while a commonly used technique in the games industry "can still feel abrupt, depending on the cues or the speed of the cross-fade" (K. Collins 2008, p. 146). Sweet also states that "effective transitions tend to be longer in length and complete a musical idea or thought", while shorter transitions "tend to sound more repetitive and less musical" (Sweet 2014, p. 167).

While the work cited above describes several criteria identified for an ideal transition in terms of rate of change, this does not seem to have been empirically evaluated in any way. This is tackled in this dissertation in Chapters 3 and 6.

2.3.4.2 Degree of Fitting

Medina-Gray (2016, p. 63) claims that "game music aesthetics tend to privilege *smoothness* – a quality in which distinct modules fit well together – as multiple modules combine"; this idea is indicative of what the ideal transition between two pieces of music should sound like.[41] However, Medina-Gray also states that "musical disjunction", where there is a lack of smoothness between two pieces of music, can also be used if dramatic circumstances call for it.

While much of Sabaneev's (1935) work discusses the composition of music for film, Kaae (2008, p. 85) states that his work "could as well have been written 70 years later as an overall instruction in how to compose dynamic music for computer games" due to his instructions on how to compose music in such a way as to be reactive to what is happening on screen.[42] In particular, Sabaneev (1935, p. 39) states that "the transition from one piece of music to another must not be casual; there must be either a modulation, or an orthodox contrast of keys". Similarly, Sabaneev (1935, p. 39) states that "the composer has to link up the individual parts and sections of his music in accordance with certain

[41]According to Medina-Gray, similar aesthetics are sought in dynamic media, such as accompanists that play music to silent films, theatre and circus performances, and DJ mixes.

[42]This is because Sabaneev discusses compositional techniques that composers could use to overcome challenges at the time when music in films was still a relatively novel idea. In particular, Sabaneev discusses the idea that music should be composed in a manner that can still work seamlessly with the film if last minute changes are needed by the director and the music may not be able to cadence successfully, particularly at a time when recording music was expensive.

musical principles, and not haphazard [sic]. He cannot discontinue a musical thought abruptly, but must at least carry it on to the cadence."

Wooller et al. (2005) discuss two different properties that are used to evaluate a successful morph between two pieces of music. These may be used in a similar manner for transitions:

smoothness
: how well does each part of the morph fit with its surrounding musical sections?

coherence
: "the logic of the musical syntax or structural form of the morph", that is: the section used for morphing should follow a similar high level structure to that found in the rest of the entire piece

Ultimately, as stated by Phillips (2014), "the key to achieving a pleasing result lies in paying very close attention to the transitions between these moments of activity, ensuring that all musical events logically rise from their circumstances and do not make overtly surprising appearances" (Phillips 2014, pp. 162–163).

In a similar manner to the rate of change category described above, the criteria associated with the degree of fitting of a transition do not seem to have been empirically evaluated. This is again tackled in this dissertation in Chapters 3 and 6.

Conclusions

This chapter has demonstrated the multiple roles that music plays in video games and how it is used differently in different parts of the game. It has also compared and contrasted different algorithmic techniques that have been used to generate music, while comparing similar work that has been done within the realm of procedural content generation in games. Finally, a closer look has been taken at the role of musical transitions at different levels of musical form and how they have been used in video games.

This chapter has also demonstrated the broadness of the term *transition* due to its musical application over various levels of hierarchical form. While the rest of this dissertation focuses on inter-piece transitions (as discussed in Section 2.3.1.3), the limitation of the term *transition* is acknowledged. Similarly, the term *piece* is normally understood to operate in what Roads (2001, pp. 3–4) calls the *macro* scale (as discussed in Section 2.3.1), but has also been used to describes music that operates on higher or lower scales of musical form.

Several areas have been identified in this chapter that will be tackled in this dissertation. First of all, certain features are claimed to be required for a good transition but have not been empirically evaluated; this is described further in Chapters 3 and 6.

This dissertation will focus on generating musical transitions between two pieces of extra-diegetic in-game music in open-world video games such as the MMORPG *World of Warcraft* (Blizzard Entertainment 2004) or the action game *Batman: Arkham Knight* (Rocksteady Studios 2015). Extra-diegetic music was chosen as it is the most common way music is presented in games, while in-game music was chosen since it has the most opportunities to change due to player interaction. Finally, open-world video games were chosen due to the way different music is assigned to various areas in the world and how the player can move between these zones, allowing for musical transitions.

This chapter has shown that two separate fields of research: algorithmic music and procedural content generation, can be brought together in order to generate music for games. By also rethinking how transitions work in video games, a novel algorithm will allow the generation of suitable musical transitions that transition between one piece of music and another. This is described further in Chapter 5. These transitions will be created using Markov models within the framework of a multiple viewpoint system in order to leverage the speed and flexibility of the models. Finally, MIDI will be used in order to take advantage of its flexibility and ease in generating dynamic music.

A symphony, like a novel, is first and foremost a dramatic narrative in which a change in relationships occurs. [...] *This means that a symphony is experienced in performance, by performers and listeners alike, not as form or structure but as a sequence of significant events in time, a drama of opposition and resolution.*

Christopher Small, *Musicking: The Meanings of Performing and Listening*, p. 159[1]

3

Investigating the Detection of Musical Transitions

This chapter details the methodology and dataset used for a study involving the perception and detection of musical transitions. The focus of the study is purely on the musical transitions and not on their interaction with the context they are placed in.

Section 3.1 describes the aims and objectives of the study in greater detail, setting out in several questions that will be answered by the study. Section 3.2 describes the

[1]Christopher Small (1998). *Musicking: The Meanings of Performing and Listening*. Middletown, Connecticut, United States of America: Wesleyan University Press, p. 232. ISBN: 9780819522573, p. 159

methodology used for the study (as well as previous iterations that were discarded, and the reasons for doing so), while Section 3.3 discusses the dataset used, and the transition techniques, included in the study. Finally, Section 3.4 describes the system design and the interface that participants interact with.

3.1 Aims and Objectives

The primary aim of this study was to investigate how easily detectable different transition techniques were when transitioning between one piece of music and another. This is motivated by Jørgensen (2008a, p. 163)'s *golden rule of audio design*, to "never let the player become annoyed or bored by the repetitiveness of the sound", and therefore would imply that an undetectable music transition is desirable. A similar implication is that an undetectable music transition is more immersive; this is explored in Chapter 6 while immersion as related to game music is discussed in Section 2.1.3.2.

Four transition techniques were chosen for this study: abrupt cut transition, horizontal resequencing, weighted averaging, and crossfading, as described in Section 2.3. A secondary aim of this study was to determine suitable descriptions and criteria that could be used to evaluate such a complex auditory phenomenon. These aims can be broken down into several different questions:

- How easily detectable are different transition techniques?
  - Is there a relationship between the evaluation criteria of a transition and how easily detectable it is?
  - Are people with musical experience better at detecting transitions?
- What are the criteria of success for a musical transition?
  - Are longer transitions more successful than shorter transitions?

– Are transitions more effective between the same source and target piece, as opposed to different source and target pieces?

3.2 Methodology

An iterative approach was taken in order to determine the best direction to take to have participants evaluate presented transitions experimentally; this approach was chosen due to the complex nature of the phenomenon being investigated. After the first couple of iterations, once it was determined just how complex and mentally taxing it was to evaluate this phenomenon, the scope of the experiment was narrowed down in order to allow for the adaptation of methodologies from music psychology.

Although not directly related to music, Bech and Zacharov (2006) provide a good general starting point for a suitable methodology for the study. In particular, according to Bech and Zacharov, for a subject to be a suitable candidate for the study, they must be considered part of the required population sample, they must not have any hearing difficulties, they must be able to successfully assess and rate the stimuli provided in the study, and they must be available for the study. Although the results of this study are intended to inform the use of transitions in a video game context, no knowledge of video games is needed for this particular study, making the viable pool of participants larger.

Two significant influences to this study's methodology include work by Deliège (1987), work done by Clarke and Krumhansl (1990), and work done by Berz and A. E. Kelly (1998). Deliège's study aims to test musical grouping rules formulated by Lerdahl and Jackendoff (1996) by using listening tests. She groups her participants into two groups, nonmusicians and musicians; however, she states that her criteria for falling into the category of musicians is their knowledge of harmony, rather than the amount of training they have for their instrument. This is because "[t]he simple practice of an instrument does not guarantee true mental training" (Deliège 1987, p. 334). This bears a resemblance to Bech and Zacharov's three-tiered distinction between untrained subjects, experienced subjects, and expert subjects.

Clarke and Krumhansl (1990) conduct six experiments in order to investigate the perception of segmentation in atonal music. Here, the authors use two pieces of music, Stockhausen's *Klavierstück IX* and Mozart's *Fantasie* in C minor, K. 475, and conduct three similar experiments for each piece, the most relevant of which is the first experiment: the identification of boundaries. Clarke and Krumhansl ask seven participants experienced in music to identify boundaries by listening to the music. First, the participants listen to the piece once. On the second playthrough of the piece, the participants are asked to identify segment boundaries by pressing a foot pedal when they hear one. On the third and final playthrough of the piece, the participants are allowed to remove any erroneous segment boundaries, but cannot add any more.

Finally, Berz and A. E. Kelly (1998) conduct a study to investigate listeners' perceptions of musical structure. Participants, grouped into inexperienced and experienced listeners, were asked to click the mouse when "a new event" began. Berz and A. E. Kelly show that inexperienced listeners were significantly worse than experienced listeners in detecting formal structural events in the music.

3.2.1 Iterative Approach

The implementation of the study was developed with reference to the above studies. Initial iterations separated rhythmic, pitch, and melodic transitions into separate experiments, the idea being to investigate whether there were differences in what constituted a smooth transition between rhythmic pieces, pitch pieces, and full melodic pieces.

3.2.1.1 First Iteration

For the first iteration of the study, participants were presented with an interface (as seen in Fig. 3.1) and briefed about the aims and objectives of the study. This version of the study was limited to investigating rhythmic transitions, and not melodic transitions, and so, no pitch information was played from any of the pieces. Each piece was presented to participants as a series of percussive rhythms taken from the dataset of pieces.

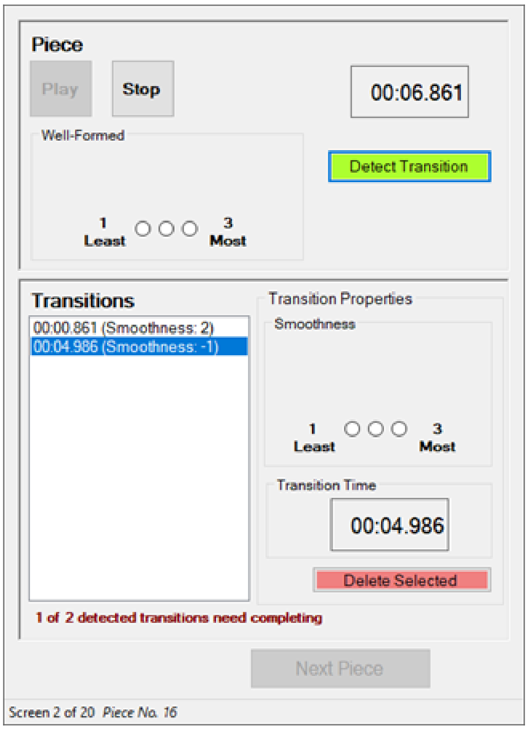


Figure 3.1: Screenshot of the first iteration of the study

The beginning and ending of a piece used in the study was marked by the use of a percussive bell sound; this was to highlight the fact that some of the pieces used in the study were not complete pieces with a proper ending, but were simply excerpts of the full piece. Participants could start and stop the currently playing piece using the given buttons, and while the piece was being played, the green *Detect Transition* button could be pressed if the participant thought that a transition was happening at that particular time in the piece. Participants were allowed to repeat the piece as many times as they liked before moving onto the next one. After the participant was satisfied with the number of transitions detected, they were asked to rate each transition for its smoothness on a 3-point scale (where 1 meant *Least smooth* and 3 meant *Most smooth*), and to rate the overall piece for how well-formed it was on a 3-point scale (where 1 meant *Not well-formed* and 3 meant *Very well-formed*).

A full record of the participant's interaction with the study was kept by logging the events and their respective timestamp. This would indicate how often the participant repeated the playback of the piece, whether or not the participant had noted any transitions which were then removed, and whether the participant changed their mind on the ratings they gave to the smoothness of the transitions and to what extent a piece was well-formed.

In total, this iteration of study contained 18 pieces, using the abrupt cut transition, horizontal resequencing, and weighted average transition techniques. Pieces were presented to participants in a randomised order.

This iteration of the study was pilot tested with 3 participants who deemed it too mentally taxing, stating that they could not cope with listening to the piece, detecting a transition, and rating that transition's smoothness at the same time. They further stated that the presented pieces were difficult and frustrating to follow due to the lack of pitch content available.

The rest of this chapter therefore discusses experiments with melodic transitions. Further iterations discussed below focus on trying to find a balance between reducing the amount of mental load for participants, as well as getting suitable transition ratings from participants.

3.2.1.2 Second Iteration

For the second iteration of the study, participants were presented with a different interface (as seen in Fig. 3.2). They were again briefed about the aims and objectives of the study, but were also exposed to dialog boxes that walked them through the same objectives. This version of the study was used to investigate melodic transitions, meaning that the melodic content of each piece (i.e. both pitch and rhythmic content) was played.

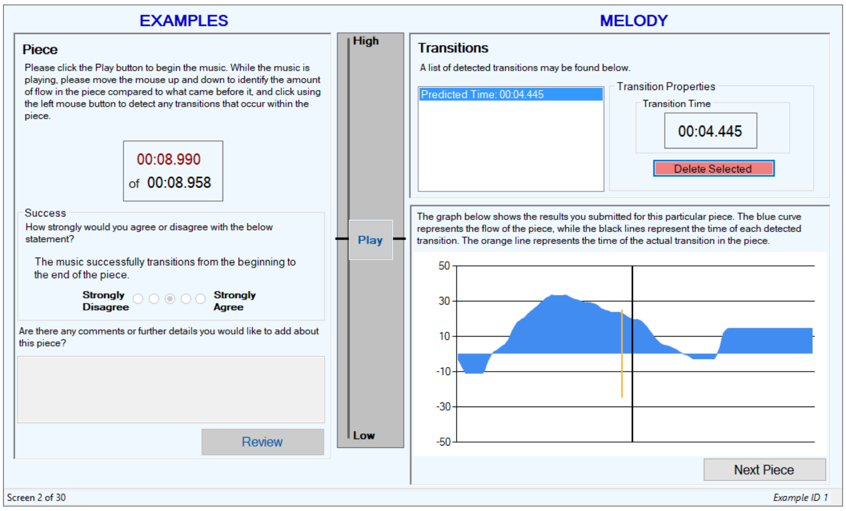


Figure 3.2: Screenshot of the second iteration of the study

There are several experimental differences between this iteration and the first. The most crucial difference is the introduction of a slider in the centre of the screen. Rather than having participants rank each detected transition on a Likert scale, they were instead asked to move the mouse up and down while the piece is playing in order to rate the smoothness of the piece in an ongoing manner, and to click the left mouse button when they felt that they had detected a transition. While the piece is playing, the position of the mouse cursor is clamped to only move freely along the displayed slider, and is released after the piece has ended. Participants were free to replay the piece as many times as they like before moving on, and to remove transitions that they believed they had detected erroneously. The ending time of the piece was also displayed, in order to give participants an idea of how long the piece was.

Upon clicking on the *Review* button to continue to the next piece, participants were shown the graph similar to the one illustrated in Fig. 3.2. This graph illustrated the smoothness curve of the piece (in blue), as well as the position of each of the participant's detected transition (in black), and the position of the actual transition (in orange).

In a similar manner to the first iteration of the study, a full record of the participant's interaction was kept. This was extended to cater for new buttons and ways of interacting with the piece. The precise data points that represent the generated smoothness curve were also stored.

In total, the study contained 28 pieces; separated into 4 example pieces that served to acquaint the user with the study interface, and 24 study pieces. Pieces were again presented to participants in a randomised order.

This iteration of the study was pilot tested with 3 participants, but was also deemed mentally taxing by users, as participants could either focus on detecting transitions by clicking the mouse button or move the mouse up and down to rate the transition smoothness, but could not do both at the same time.

3.2.2 Third Iteration

In this iteration, participants were presented with a study consisting of 4 example transition pieces and 18 study transition pieces. A screenshot of the user interface of this iteration can be seen below in Fig. 3.3. In a similar manner to Clarke and Krumhansl (1990) and Berz and A. E. Kelly (1998), each participant listened to the respective piece once and without any interaction. On their second and final listen, they were then asked

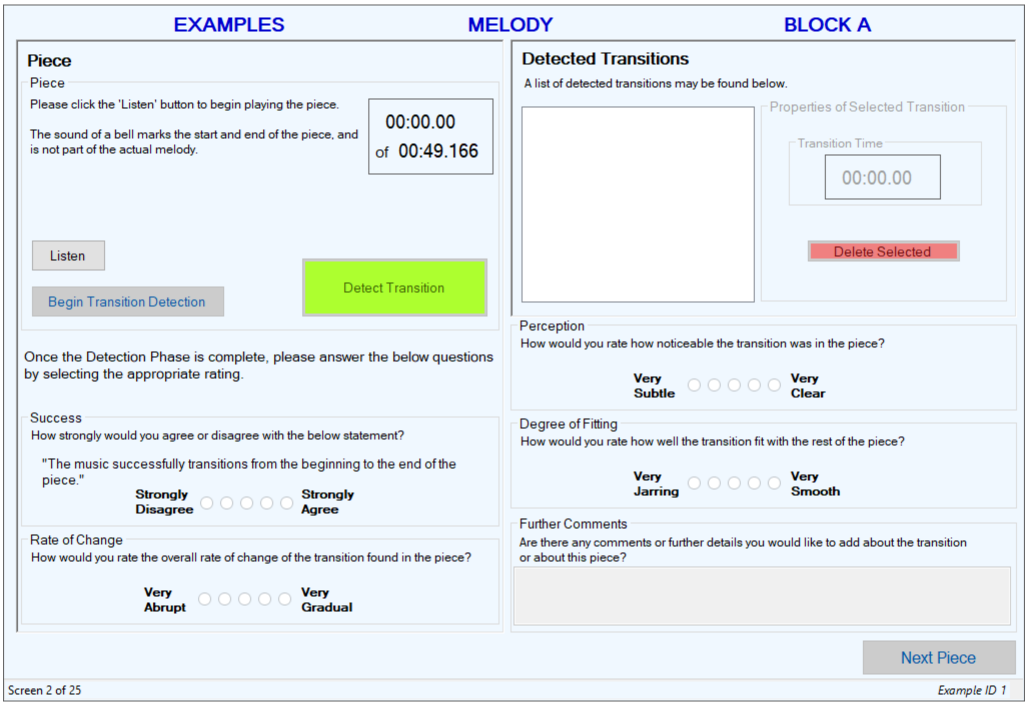


Figure 3.3: Screenshot of the third and final iteration of the study

to click on the green *Detect Transition* button when they thought that they had detected a transition in the piece.

In a similar manner to the second iteration, participants were allowed to detect multiple transitions in the same piece (since the instructions given to them do not state how many possible transitions there are), and could remove erroneous detections afterwards. However, they were not allowed to add new ones after the second and final listening. Finally, they were asked to rate the piece based on the success of its transition, and to rate each detected transition based on four criteria, detailed below. Participants could also add further comments in a text area if they wished to do so.

- **Success**: How strongly would you agree or disagree with the below statement? – "The music successfully transitions from the beginning to the end of the piece."

  - Participants were given a 5-point scale ranging from *Strongly Disagree* to *Strongly Agree*

- **Rate of Change**: How would you rate the overall rate of change of the transition found in the piece?
  - Participants were given a 5-point scale ranging from *Very Abrupt* to *Very Gradual*

- **Perception**: How would you rate how noticeable the transition was in the piece?
  - Participants were given a 5-point scale ranging from *Very Subtle* to *Very Clear*

- **Degree of Fitting**: How would you rate how well the transition fit with the rest of the piece?
  - Participants were given a 5-point scale ranging from *Very Jarring* to *Very Smooth*

Therefore, a high value for *Success* indicates that the participant strongly agrees that the music successfully transitioned from the beginning to the end of the presented music, while a low value shows that the participant strongly disagrees with that statement. In a similar manner, a high value for *Rate of Change* shows that the participant found the music to transition very gradually, while a low value indicates that the participant found the music to transition very abruptly. A high value for *Perception* shows that the participant found the music to transition very clearly and noticeably, while a low value shows that the participant found the music to transition in a very subtle manner. Finally, a high value for *Degree of Fitting* indicates that the participant found the transition to be very smooth, while a low value indicates that the participant found the transition to be very jarring.

At the end of the study, participants were asked to answer a few questions related to their demographics, as can be seen in Fig. 3.4.

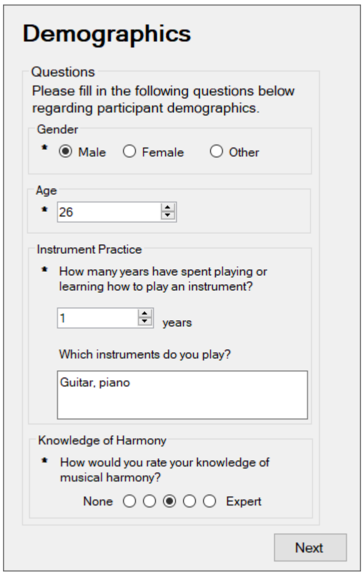


Figure 3.4: Screenshot of the demographics screen in the third and final iteration of the study

3.3 Dataset

Six pieces were chosen to be used in the main study: two marches taken from classical music, and four pieces of music taken from video games.[2] Eight original pieces were also selected to create four example transition pieces, each using a different transition technique. All chosen pieces (both for the study and as examples) were in common time, but had different tempo markings and key signatures.

3.3.1 Example Pieces

Four example transition pieces were created to be used as examples shown to each participant before interacting with the main body of the study. These are shown below:

- two are countries' national anthems
  - Germany's *Deutschlandlied* (Haydn 1797)

[2] As discussed in Section 2.1.3.3, video game music cannot really be classified as one particular genre, but rather as making use of multiple different genres.

  - Australia's *Advance Australia Fair* (McCormick n.d.)

- four pieces are traditional Christmas carols

  - While Shepherds Watched Their Flocks by Night (Rondeau 2009)
  - O Come, All Ye Faithful (Wade 1751)
  - Angels We Have Heard on High (Barnes n.d.)
  - God Rest Ye Merry Gentlemen (Hudson n.d.)

- one piece is an arrangement of a traditional sea shanty

  - The Drunken Sailor (Strohbach 1963)

- one piece is a prelude to most games in the *Final Fantasy* series, and is composed entirely of arpeggios

  - Final Fantasy - Prelude (Gluck n.d.)

The tempo and key signatures for each piece can be seen in Table 3.1.

Tempo	**Key Signature**	**Piece**
120	E♭ major	*Deutschlandlied* (Haydn 1797)
N/A	C major	*Advance Australia Fair* (McCormick n.d.)
120	F major	*While Shepherds Watched Their Flocks by Night* (Rondeau 2009)
N/A	G major	*O Come, All Ye Faithful* (Wade 1751)
N/A	F major	*Angels We Have Heard on High* (Barnes n.d.)
N/A	E minor	*God Rest Ye Merry Gentlemen* (Hudson n.d.)
120	G major	*The Drunken Sailor* (Strohbach 1963)
160	A minor	*Final Fantasy - Prelude* (Gluck n.d.)

Table 3.1: Example piece selection for study

The above pieces were then combined to form four example transition pieces, with each piece using a different transition technique. These example transition pieces can be seen below in Table 3.2.

Example Transition Piece (Source to Target)	**Transition Type**
Final Fantasy - Prelude to *God Rest Ye Merry Gentlemen*	Abrupt Cut
O Come All Ye Faithful to *Deutschlandlied*	Horizontal Resequencing
While Shepherds Watched Their Flocks to *Drunken Sailor*	Weighted Averaging
Advance Australia Fair to *Angels We Have Heard On High*	Crossfade

Table 3.2: Example pieces

3.3.2 Study Pieces

The six pieces chosen for the main part of the study can be seen below in Table 3.3.

Type	**Tempo**	**Key Signature**	**Piece**
March	N/A	D major	*Radetzky March, Op. 228*, by Johann Strauss (Strauss n.d.)
March	144	G major	*March of the Toy Soldiers, The Nutcracker Suite, Op. 71*, by Pyotr Tchaikovsky (Tchaikovsky 1892)
Game	117	G major	*The Pantheon (Ain't Gonna Catch You)*, by Darren Korb, written for *Bastion* (Korb 2012)
Game	110	G major	*Rise*, by Raney Shockne and transcribed by Patti Rudisill, written for *Dragon Age: Inquisition* (Shockne 2015)
Game	108	F major	*Rikku's Theme*, by Junya Nakano, Masashi Hamauzu, & Nobuo Uematsu, and arranged by Masashi Hamauzu, written for *Final Fantasy X* (Square Enix 2002)
Game	92	D major	*Eyes on Me*, by Nobuo Uematsu, and arranged by Shirō Hamaguchi, written for *Final Fantasy VIII* (Square Enix 2000)

Table 3.3: Piece selection for the main study

One of the main decisions taken for the above choice of music used was the idea to use music that could realistically be used in a video game. For example, classical music was often used in early computer games to avoid restrictive music licensing agreements, and also because of the technical limitations of the audio hardware at the time (Gibbons 2009); for example, Tchaikovsky's *March of the Toy Soldiers* was used in the platformer *Dynamite Headdy* (Treasure 1994), and Strauss's *Radetzky March* was used in the platformer *Technician Ted* (Marsden and Cooke 1984).

Furthermore, while a vast amount of music has been composed for games, very little of this is accessible to the public in the form of officially released sheet music. When this is done, the selection of scored music is usually restricted to popular pieces from games such as prominent musical themes for main characters or locations in games. While fan-made transcriptions exist of most compositions, a conscious decision was made to only use sheet music (both digital or otherwise) that has been officially released by the game's publisher.

Thus, the six chosen pieces represent a mixture of classical and modern music, vocal and non-vocal music, diegetic and extra-diegetic music, and thematic and non-thematic music, with varying degrees of instrumentation, composers, game genres, and moods used in the pieces. While it is assumed that the reader is familiar with both Tchaikovsky's *March of the Toy Soldiers* and Strauss's *Radetzky March*, more detail is given below for the other 4 pieces chosen with regards to their game's original context.

Most music written for *Final Fantasy* games normally do not have lyrics as the pieces are intended to be used as thematic background music, fitting appropriately to both the mood of the narrative as well as the in-game location. As JRPGs, music written for *Final Fantasy* games tends to be highly motific, with most main characters and in-game locations have their own themes. A good example is *Rikku's Theme*, a character theme from *Final Fantasy X* (Square 2001). This theme plays when one of the main characters of the game, Rikku, has a prominent role in the game's narrative, as is itself a variant of another piece written for the game, *Oui are Al Bhed*, with the Al Bhed being a different ethnicity in the game world that Rikku is a part of. In contrast however, *Eyes On Me* is one of the main themes in *Final Fantasy VIII* (Square 1999); it is a ballad with lyrics written by Nobuo Uematsu for Chinese singer Faye Wong. The piece plays a heavy narrative role in the game's storyline, and appears towards the end of the game, with an instrumental arrangement titled *Julia* and a waltz arrangement titled *Waltz for the Moon*, both appearing towards the beginning of the game.

The Pantheon (*Ain't Gonna Catch You*) is a piece featured in *Bastion* (Supergiant Games 2011), an action game with RPG elements. The piece can be found as bonus material towards the end of game once certain conditions are met. While the in-game piece features a vocalist and a guitar, sheet music for both guitar and piano are provided, accompanied by a vocal melody. *Rise* is a tavern song taken from the RPG *Dragon Age: Inquisition* (BioWare 2014), and the only example of diegetic music from the choice of video game music in the study, as the song is played by in-game bards that can be found in taverns in the game. While the song contains vocals and guitars, the sheet music provided by the publisher only contains the vocal melody, which is what was used for the study.

While it is assumed that some participants may not have prior familiarity with the game music chosen for the study, a larger number of participants may have prior familiarity with the classical pieces chosen. This implies that if participants hear notes that do not match with their familiarity of the piece, they can assume that the transition has most likely taken place. While participants are likely to get more and more familiar with the pieces (including ones with no prior familiarity) as they progress through the study, this is not considered to have a major effect on the study, since each participant is presented with a unique combination of source and target piece, and each piece is heard for a limited amount of times.

3.3.3 Musical Representation

The musical representation used for this study can be seen below in Fig. 3.5, with each music event represented by a duration value and a pitch value.

```
8D5, 8R, 12D5, 12D5, 12D5, 8E5, 8R, 8E5, 8R, |, 8F♯5, 8R,
8D5, 8R, 2E5, |, ...
```

Figure 3.5: A snippet of the musical representation chosen for pieces

Duration values for each musical event were represented in a similar manner to the *Humdrum **kern representation for music notation* (Huron 1998)[3], where `1` represents a semibreve, `8` represents a quaver, and a `16.` represents a dotted semiquaver. This is then internally converted to a numeric representation based on a granularity value of `96` taken from M. Pearce et al. (2005), where a semibreve is represented by `96`, a quaver is represented by `12`, and a dotted semiquaver is represented by `9`. A full table of this representation can be seen in Appendix F.

Pitches are represented as a combination of a letter representation of the note's pitch along with any required accidentals ♯, ♭, and ♮[4], along with a numeric representation of the octave that the note is played on (Young 1939). Rests were simply notated by using the letter `R`.

Where necessary, each barline was noted by using a `|` symbol, and the beginning and end of a transition were noted by using the `£` and `€` symbols respectively. Phrase boundaries were represented using the characters < and > to signify the beginning and end of a phrase respectively. Any sharps or flats from the piece's key signature were applied to each individual note in the entire piece in the representation, thus removing the key signature entirely. Any performance markings were ignored, including tie markings; the latter to ensure certain transition techniques worked properly. One such example is

[3]An updated version of this musical representation can be found at `http://www.humdrum.org/rep/kern/`

[4]Enharmonic equivalence was used while processing pieces in order to have a consistent representation for Markov models to use. This means for example, that `4B♭4` would instead be represented internally as `4A♯4`.

when using horizontal resequencing, since the target piece is played at the end of the next bar when an event is triggered, which could potentially break tied notes.

Since crossfading transitions could potentially have two pieces play at the same time, each full piece was represented by two distinct parts: the source piece that would eventually fade out, and the target piece that would eventually fade in. Both parts would be scheduled accordingly so that the output would play at the same time. For crossfading transitions, a velocity parameter[5] was added to notes by first adding a `%` symbol to distinguish from any melodic content. The amount of velocity needed was then added by using the appropriate numeric amount and the character `V`; thus, `%40V` represents a velocity value of 40. An example of a crossfading transition is shown below in Fig. 3.6.

First: `4E5, 4D5, 4C5, 4B4%80V, 4A4%60V, 4G4%40V, 4F4%20V, 4R, 4R`

Second: `4R, 4R, 4R, 4E4%20V, 4F4%40V, 4G4%60V, 4A4%80V, 4B4, 4C5`

Figure 3.6: An example of the representation used to play a piece containing a crossfade transition

3.3.4 Dataset Configuration for the Study

To conduct a thorough study of the phenomenon in question, a 288 transition piece dataset was created for the study by taking all possible combinations of the six study pieces as source pieces and target pieces, the four chosen transition techniques, and the two different transition positions. All pre-processing, such as generating the dataset and dividing it into batches, was done offline before the study took place in order to preserve consistency. When generating transitions between two pieces for the dataset, a suitable

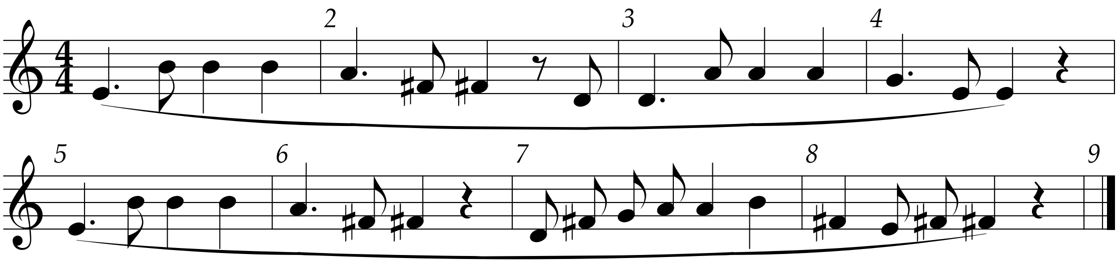
[5]As per the General MIDI specification (*General MIDI System Level 1* 1991), velocity is a numeric value ranging from 0 to 127. A value of 100 was taken as the default velocity value of notes to be played by the system. Note that while velocity is normally interpreted to be the dynamic level or loudness of individual notes, the specifications themselves do not specify a particular interpretation (Dannenberg 2006)

position was randomly selected in the source piece, making sure that enough of the piece was available in order to provide the participant with enough musical context.

3.3.4.1 Transition Positions

One variable introduced to the study was the position at which the transition took place. This could be at one of two locations: within a musical phrase, or outside a musical phrase. A transition is determined to be a within-phrase transition if the event is triggered while the music is currently within a phrase. Similarly, a transition is an outside-of-phrase transition if the event is triggered between one phrase and another.

Example 3.1 shows an extract of *Rise* from *Dragon Age: Inquisition*, with phrase markings added by the author for convenience. Here, a within-phrase transition is one that is triggered within the phrases shown in the extract (i.e. at any time within the phrase shown in bars 1-4, and within the phrase shown in bars 5-8), while an outside of phrase transition is one triggered between phrases (i.e. at bar 4 or bar 8, before or after each crotchet rest).



Example 3.1: A musical extract from *Rise* from *Dragon Age: Inquisition*. Phrase markings added by the author.

3.3.4.2 Transition Techniques

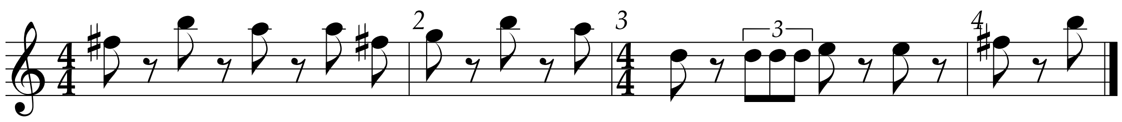
For this study, four transition techniques were chosen: the abrupt cut transition, horizontal resequencing, crossfading, and weighted averaging. The abrupt cut transition is frequently claimed in the literature to be a poor performing transition technique, and thus serves as a good comparison between the other techniques. Horizontal resequencing and crossfading are both commonly used in the industry, with the crossfading technique being the most frequently used due to its ease of implementation. Finally, the weighted averaging technique was chosen due to its potential application in generative transitions for games. While a detailed description of each transition technique was given in Section 2.3, this section will focus on the specific representation and implementation of each technique.

3.3.4.2.1 Abrupt Cut

Fig. 3.7a shows an extract from a piece in the dataset to demonstrate how an abrupt cut transition technique is represented in the data set. Example 3.7b visualises this extract using music notation, where the abrupt cut transition is represented in bar 3 by reintroducing the time signature. Note the incomplete bar in bar 2; this is because the glsnom:abruptcuttransition occurred before the bar was over.

```
8F♯5, 8R, 8B5, 8R, 8A5, >, 8R, <, 8A5, 8F♯5, |, 8G5, 8R,
8B5, 8R, 8A5, 8R, 8A5, £, <, 8D5, 8R, 12D5, 12D5, 12D5,
8E5, 8R, 8E5, 8R, |, 8F♯5, 8R, 8D5
```

(a) A representation of the resulting transition



(b) A musical extract of the resulting transition

Figure 3.7: A within-phrase abrupt cut transition between *Radetzky March* and *March of the Toy Soldiers*

3.3.4.2.2 Horizontal Resequencing

A horizontal resequencing transition technique can be seen in the musical representation in Fig. 3.8a; the corresponding musical example 3.8b visualises this extract using music notation. In a similar manner to the abrupt cut transition discussed in Section 3.3.4.2.1, the horizontal resequencing transition is represented in bar 3 by reintroducing the time signature.[6]

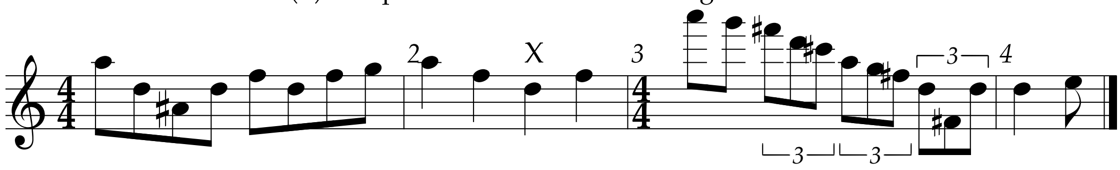
As stated at the beginning of Section 3.3.4, a suitable position was randomly selected in the source piece in order to determine where the transition should take place. While no further changes are made with an abrupt cut transition, this position is moved to the end of the chosen bar in a horizontal resequencing transition.

Note that while in an interactive environment there is some sort of event that triggers the need for a horizontal resequencing transition (an example of this is shown in 3.8b marked by the **X** symbol), this cannot be replicated in this particular study due to the lack of an interactive environment.

[6]Note that in a similar manner to an abrupt cut transition, no new musical material is generated here. This is because when an appropriate in-game event is received that triggers a horizontal resequencing transition, the music is allowed to reach the end of the bar before the target piece begins playing, as discussed earlier in Section 2.3.3.4

```
8A5, 8D5, 8A♯4, 8D5, 8F5, 8D5, 8F5, 8G5, |, 4A5, 4F5, 4D5,
4F5, >, |, £, <, 8A6, 8G6, 12F♯6, 12D6, 12C♯6, 12A5, 12G5,
12F♯5, 12D5, 12F♯4, 12D5, >, |, <, 4D5, 8E5
```

(a) A representation of the resulting transition



(b) A musical extract of the resulting transition

Figure 3.8: A horizontal resequencing outside of phrase transition between *Rikku's Theme* from *Final Fantasy X* and *Eyes on Me* from *Final Fantasy VIII*

3.3.4.2.3 Weighted Averaging

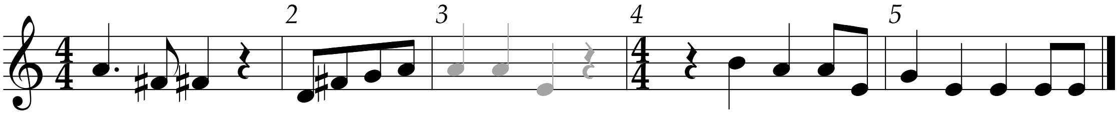
Fig. 3.9a shows the representation of a weighted averaging transition technique, with the corresponding musical example illustrated in Example 3.9b. The musical material generated for the transition can be seen in bar 3 in a grey colour.

The weighted averaging transition is performed by using first-order Markov models, with the models using a pitch-duration representation (as shown earlier in Fig. 3.5. Bar lines (|), phrase markings (< and >), and transition markings (£ and €) were ignored by the Markov models. A step count of 4 was used, meaning that generated transitions would always be 4 musical events long, regardless of the duration of the event.[7]

[7]Of course, this ultimately depends on the pieces used as input for the Markov models, but theoretically, a transition generated using this technique could be made up of a segment as short as 4 semiquavers, or as long as 4 semibreves.

```
4.A4, 8F♯4, 4F♯4, 4R, |, 8D4, 8F♯4, 8G4, 8A4, £, 4A4, 4A4,
4E4, 4R, €, 4R, <, 4B4, 4A4, 8A4, 8E4, |, 4G4, 4E4, 4E4,
8E4, 8E4
```

(a) A representation of the resulting transition.



(b) A musical extract of the resulting transition.

Figure 3.9: A weighted averaging within-phrase transition between *Rise* and *The Pantheon*

3.3.4.2.4 Crossfading

Finally, a crossfade transition technique can be seen represented in Fig. 3.10a, and as the corresponding musical example as seen in Extract 3.10b. Here, the transition was implemented as both parts fading in and fading out at the same time (rather than fading to silence and subsequently fading from silence). Each crossfading transition was also hard-coded to last for 10 musical events, whether or not they were rests or notes, and each musical event increased or decreased the velocity by a value of 10.

As can be seen in the figures below, the target piece (i.e. the part which is fading in) is initially represented with rests of a value equivalent to the musical events happening in the first part of the piece. After the crossfade is complete, the source piece has faded out and therefore stops playing.

First: 8E5, 16R, 16D5, 8C5, 16R, 16E5, 8F♯4, 16R, 16E5, 8D5, 16R, 16F♯4, |, £, 8G5%90V, 16R%80V, 16F♯5%70V, 8E5%60V, 16R%50V, 16F♯5%40V, 8D♯5%30V, 8R%20V, 8B5%10V

Second: 8R, 16R, 16R, 8R, 16R, 16R, 8R, 16R, 16R, 16R, 16R, 8R, <, 8A6%10V, 8G6%20V, 12F♯6%30V, 12D6%40V, 12C♯6%50V, 12A5%60V, 12G5%70V, 12F♯5%80V, 12D5%90V, €, 12F♯4, 12D5, >, |, <, 4D5, 8E5, 8F♯5, 4E5, 4C♯5

(a) A representation of the resulting transition.

(b) A musical extract of the resulting transition.

Figure 3.10: A crossfade outside of phase transition between *March of the Toy Soldiers* and *Eyes on Me* from *Final Fantasy VIII*

3.3.4.3 Batches

As explained earlier, a 288 piece dataset was created for the study by taking into account all possible combinations of source and target pieces, transition techniques, and transition positions. This dataset was then split up into 18 different batches, balanced in such a way as to have each batch contain an equal amount of pieces with within-phrase and outside of phrase transitions, and similar amounts of pieces with different transition techniques. Complete coverage of the entire dataset was ensured by having 32 participants, meaning that the entire dataset would be covered twice. Each study participant was presented with a particular batch in a random order.

3.4 System Design

The system used for this preliminary study was coded in C# using WinForms, and the *midi-dot-net* library[8] used to schedule MIDI sounds in an internal message queue. By default, pitch sounds were played over Channel 1, but Channel 2 was also used when using the crossfade transition technique, with both channels playing at the same time.

Percussion sounds for initial iterations of the study were assigned to the *Snare Drum 1* sound and were played over Channel 38, while the default *Acoustic Grand Piano* sound, assigned to instrument ID 0 according to the General MIDI specification (*General MIDI System Level 1* 1991), was used for Channels 1 and 2. The *Microsoft GS Wavetable SW Synth*, the default software synthesizer found on Windows, was used as the software instrument for this study. Each piece in the study was played to participants at a hard-coded tempo of 120 beats per minute, regardless of the piece's original tempo.

[8] While the original version of the *midi-dot-net* library was created by Tom Lokovic (which can be found at `https://code.google.com/archive/p/midi-dot-net/`), this study uses a fork by Justin Ryan, which can be found at `https://github.com/jstnryan/midi-dot-net`.

3.5 Conclusions

This chapter has presented a study framework aimed to investigate how easily detectable four different musical transition techniques are: abrupt cut transition, horizontal resequencing, crossfading, and weighted averaging. The technique used in this study, where participants must click a mouse button when they have detected a musical transition within the piece, has been adapted from methodologies used in music psychology studies such as Deliège (1987) and Clarke and Krumhansl (1990). The transitions are then rated based on their success, their rate of change, their perception, and their degree of fitting within the piece. The following chapter analyses and discusses the results of this study.

Thus it is that music, mirroring the essential shape and substance of human experience, from time to time contains sudden, shocking clashes with unpredictable chance. Lesser composers tend to eschew such harsh encounters with the unexpected, avoiding them by employing a single-minded sameness of musical materials or minimizing them by making a fetish of well-oiled, smooth transitions. But the great masters have faced fate boldly, and capricious clashes with chance are present in much of their finest music.

Leonard Meyer, *Emotion and Meaning in Music*, p. 196[1]

4

Evaluating the Detection of Musical Transitions

This chapter describes the techniques used to evaluate the study described in Chapter 3. Section 4.1 goes into more detail about the participants that took part in this study. Section 4.2 describes the use of non-parametric tests in order to evaluate the obtained results, while Section 4.3 discusses the observations made from the results of the study.

[1]Leonard B. Meyer (1961). *Emotion and Meaning in Music*. University of Chicago Press, p. 320. ISBN: 978-0226521398, p. 196

4.1 Participants

Each participant was asked to enter their demographics at the end of the study; a screenshot of the provided form can be seen in Fig. 4.1. 32 participants completed the study, split between 18 males and 14 females, and were recruited via word of mouth, forming a convenience sample. The age of the participants ranged from 18–63 years old (mean age: 35.5, s.d. = 13.9). The vast majority of participants were Maltese (28), while the remaining 4 were of Greek, Polish, Norwegian, and Icelandic origin. All participants had a Western background and upbringing.

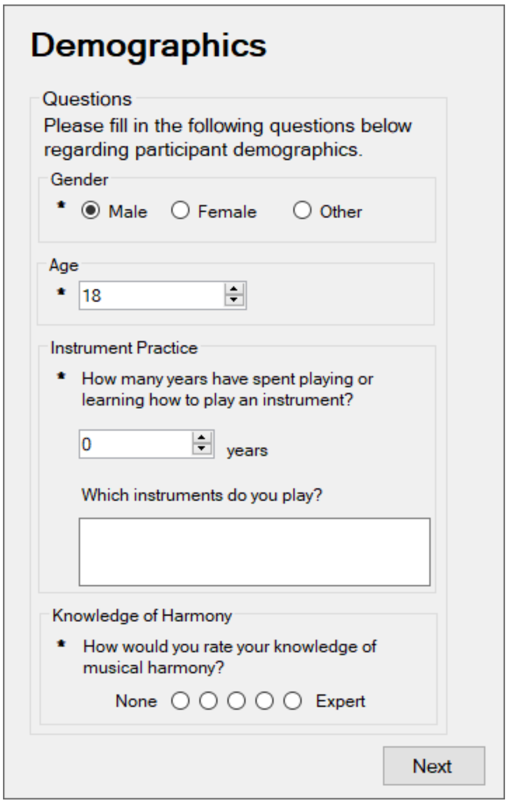


Figure 4.1: Screenshot of the demographics collection screen from the study

Participants were asked to self-rate their knowledge of musical harmony according to the 5-point scale presented in Fig. 4.1. This means that only the first and last points of the scale were labelled (with the labels *None* and *Expert* respectively). The following results were obtained:

- none (8)
- minimal (8)
- some (11)
- knowledgeable (3)
- expert (2)

The results above were grouped into 16 untrained participants (made up of participants rated as *none* and *minimal*) and 16 trained participants (made up of participants rated as *some*, *knowledgeable*, and *expert*).

Participants were also asked how many years they had spent practising an instrument. 10 participants responded that they had never practised an instrument before, with the remaining 22 participants ranging from 1–50 years of instrument practice (mean: 12.36, s.d. = 13.77). Out of these 22, 2 participants did not list which instruments they played, while the rest reported experience with 1-4 instruments (mean count: 1.75, s.d. = 0.89). 22 participants expressed experience with a harmonic instrument, 24 participants with a melodic instrument, and 2 with a rhythmic instrument (taking into consideration that some participants expressed experience with more than one instrument type, and some instruments can be classified under multiple categories). Table 4.1 shows the instruments played by this study's participants and their categorisation, and was verified by an expert.

Two of the participants did not click the mouse button to detect transitions for the entire study, while one participant only did so for the examples. This means that for these participants, the presented pieces only contain the ratings for success, rate of change, perception, and degree of fitting, while missing any timestamps for transition detections.

Instrument	Category	Participant Count
Cello	Melodic	2
Clarinet	Melodic	1
Drums	Rhythmic	2
Flute	Melodic	1
Guitar	Harmonic[2]	8
Mandolin	Melodic	1
Ocarina	Melodic	1
Piano	Melodic/Harmonic	12
Pipe Organ	Melodic/Harmonic	1
Ukulele	Harmonic	1
Viola	Melodic	1
Violin	Melodic	3
Voice	Melodic	1

Table 4.1: Categorisation of the instruments played by participants

4.2 Non-Parametric Tests

The section discusses the use of non-parametric tests on the data collected for this study. As stated in Section 3.2.2, the data associated with a transition piece included the timestamps of when participants detected transitions within the piece, as well as the ratings of the transition's success, rate of change, perception, and degree of fitting values. Each of the latter values were taken from a 5-point Likert scale.

4.2.1 Using Discrete Visual Analogue Scales

Each criterion is presented to participants as a 5-point discrete visual analogue scale. As stated by Uebersax (2006), this type of scale presents participants with 5 categories that they can use to answer the question. All categories are unlabelled except for the extremes, and only the categories presented in the scale may be used.[3] Data from a discrete visual

[2] No participants reported having experience with playing classical guitar, so all guitar players were grouped together under the *harmonic* category.

[3] This is in contrast to a visual analogue scale, where participants can place a mark anywhere along a line

analogue scale must be treated as categorical data and not interval data; this is because participants may perceive different distances between each category on the scale (Brill 2008). This is in contrast to items in a Likert scale, where each point on the scale must be more or less evenly spaced and notated with consecutive numbers and verbal labels (Uebersax 2006).

Discrete visual analogue scales result in ordinal values. These do not have a normal distribution, meaning that non-parametric tests must be used for statistical analysis.

4.2.2 Issues and biases with self-reporting and scales

Although participants were asked how many years they spent playing or learning how to play an instrument, no definition was provided as to what this meant. This meant that there was no difference between participants that took formal training in an instrument, or participants that were self-taught. Furthermore, just because a participant stated that they have been playing an instrument for 10 years does not necessarily mean that there was improvement to that participant's skill in playing that instrument. Similarly, with participants that reported playing multiple instruments, there was no granular detail asked as to how many years were spent playing each instrument.

A similar issue can be found when asking participants to rate their knowledge of harmony; no definition is given for this in the question provided, and therefore the question may be interpreted in different ways.

Several biases as discussed by Brill (2008) when using self-reporting techniques such as Likert items. One bias is the *central tendency bias*, where participants are more inclined to avoid values on both extremes of a scale. Other biases include *acquiescence bias* (where participants tend to choose answers that they believe is being expected of them) and *social desirability bias* (where participants select answers that portray themselves in a favourable light). None of these biases are deemed applicable to the ratings of each transition technique, since the question is about the technique itself rather than the participant.

4.2.3 Techniques

All statistical tests described below were carried out in statistical package R (R Core Team 2017).

A linear-by-linear test[4] (also known as an ordinal chi-squared test) was used to determine association between two variables in a contingency table (Agresti 2007, pp. 229–232), with the null hypothesis being that there existed no association between them.

The list below shows the results for tests run for each pair of Likert scale variables (success, rate of change, perception, and degree of fitting, as discussed in Section 3.2.2), presenting the corresponding χ^2 value, p-value and degree of freedom value for each test. The α value is taken to be 0.05, and if the p-value is lower than α, the null hypothesis is rejected. Cramér's V (also referred to as Cramér's ϕ) was also computed to measure the effect size of each association, where a small effect size has a magnitude of 0.1, a medium effect size has a magnitude of 0.3, and a large effect size has a magnitude of 0.5 (Cohen 1988, pp. 223–226).

Pair of Variables	χ^2	**df**	**p-value**	**Cramer's V**	**Effect size**
Success vs. Rate of Change	443.16	16	$p < 2.2 \times 10^{-16}$	0.397	medium
Success vs. Perception	275.56	16	$p < 2.2 \times 10^{-16}$	0.313	medium
Success vs. Degree of Fitting	665.86	16	$p < 2.2 \times 10^{-16}$	0.486	medium
Rate of Change vs. Perception	457.12	16	$p < 2.2 \times 10^{-16}$	0.403	medium
Rate of Change vs. Degree of Fitting	687.75	16	$p < 2.2 \times 10^{-16}$	0.494	medium
Perception vs. Degree of Fitting	447.56	16	$p < 2.2 \times 10^{-16}$	0.399	medium

Table 4.2: Results from the ordinal chi-squared test for each pair of Likert variables

The above results in Table 4.2 show that the null hypothesis is rejected in all cases, meaning that there is an association between each pair of variables.

[4] `lbl_test` was the function used to run this test in R Studio, taken from the `coin` package

Kendall's tau is used to determine whether the relationship between two variables is monotonic (i.e. as one variable increases in value, the other variable also increases) and measured on a scale of -1 to 1. This is calculated for each statistically significant pairwise difference, and the positive correlations found can be seen below in Table 4.3.

Pair of Variables	Z	p-value	Kendall's tau	Effect size
Success vs. Rate of Change	14.37	$p < 2.2 \times 10^{-16}$	0.44	medium
Success vs. Degree of Fitting	19.153	$p < 2.2 \times 10^{-16}$	0.587	medium
Rate of Change vs. Degree of Fitting	20.321	$p < 2.2 \times 10^{-16}$	0.62	medium

Table 4.3: Positive correlations found for each pair of Likert variables

Negative correlations that were also found between the pairwise differences are shown below in Table 4.4.

Pair of Variables	Z	p-value	Kendall's tau	Effect size
Success vs. Perception	-8.53	$p < 2.2 \times 10^{-16}$	-0.261	weak
Rate of Change vs. Perception	-12.699	$p < 2.2 \times 10^{-16}$	-0.388	medium
Perception vs. Degree of Fitting	-13.543	$p < 2.2 \times 10^{-16}$	-0.414	medium

Table 4.4: Negative correlations found for each pair of Likert variables

Taking into consideration the value ranges associated with each variable (as discussed in Section 3.2.2), these results demonstrate that:

- A transition can be considered successful if the rate of change between one piece and another is gradual (due to a medium strength positive correlation between Success and Rate of Change).

- A transition can be considered successful if the degree of fitting between one piece and another is very smooth (due to a medium strength positive correlation between Success and Degree of Fitting).

- A transition can be considered successful if the change between one piece and another is perceived to be very subtle (due to a medium strength negative correlation between Success and Perception).

- A subtle transition is one that has a high degree of fitting between one piece and another (due to a medium strength negative correlation between Perception and Degree of Fitting).
- A gradual transition is perceived to be very subtle (due to a medium strength negative correlation between Rate of Change and Perception).
- A gradual transition is one that has a high degree of fitting between one piece and another (due to a medium strength positive correlation between Rate of Change and Degree of Fitting).

These results confirm what is stated in the literature, as can be seen in Section 2.3.4.

The four transition techniques were grouped into 2 categories based on the resulting transition length: the abrupt cut transition and horizontal resequencing transition techniques were placed in the *Short* category, while the weighted averaging and crossfading techniques were placed in the *Long* category. The Wilcoxon Rank Sum Test (also known as the Mann-Whitney U Test) was used to compare the differences between means of these two categories, with the null hypothesis being that there is no difference in ratings between both categories. This test was also used to compare the differences between the means of transition position (that is, whether or not a transition was either within-phrase or outside-of-phrase). The obtained results are shown below in Table 4.5.

	Transition Length, p	**Transition Position,** p
Success	$\mathbf{0.00797}$	0.386
Rate of Change	$\mathbf{0.0428}$	0.0642
Perception	$\mathbf{0.00101}$	0.0936
Degree of Fitting	$\mathbf{2.005 \times 10^{-5}}$	$\mathbf{0.0373}$

Table 4.5: Results of the Wilcoxon Rank Sum Test for transition length

The results obtained for transition length categories show are all statistically significant with a p-value < 0.05, meaning that there is a difference between the 2 categories.

Fig 4.2 below shows that short transitions seem to have been rated as more successful, smoother, subtler, and more gradual than longer transitions.

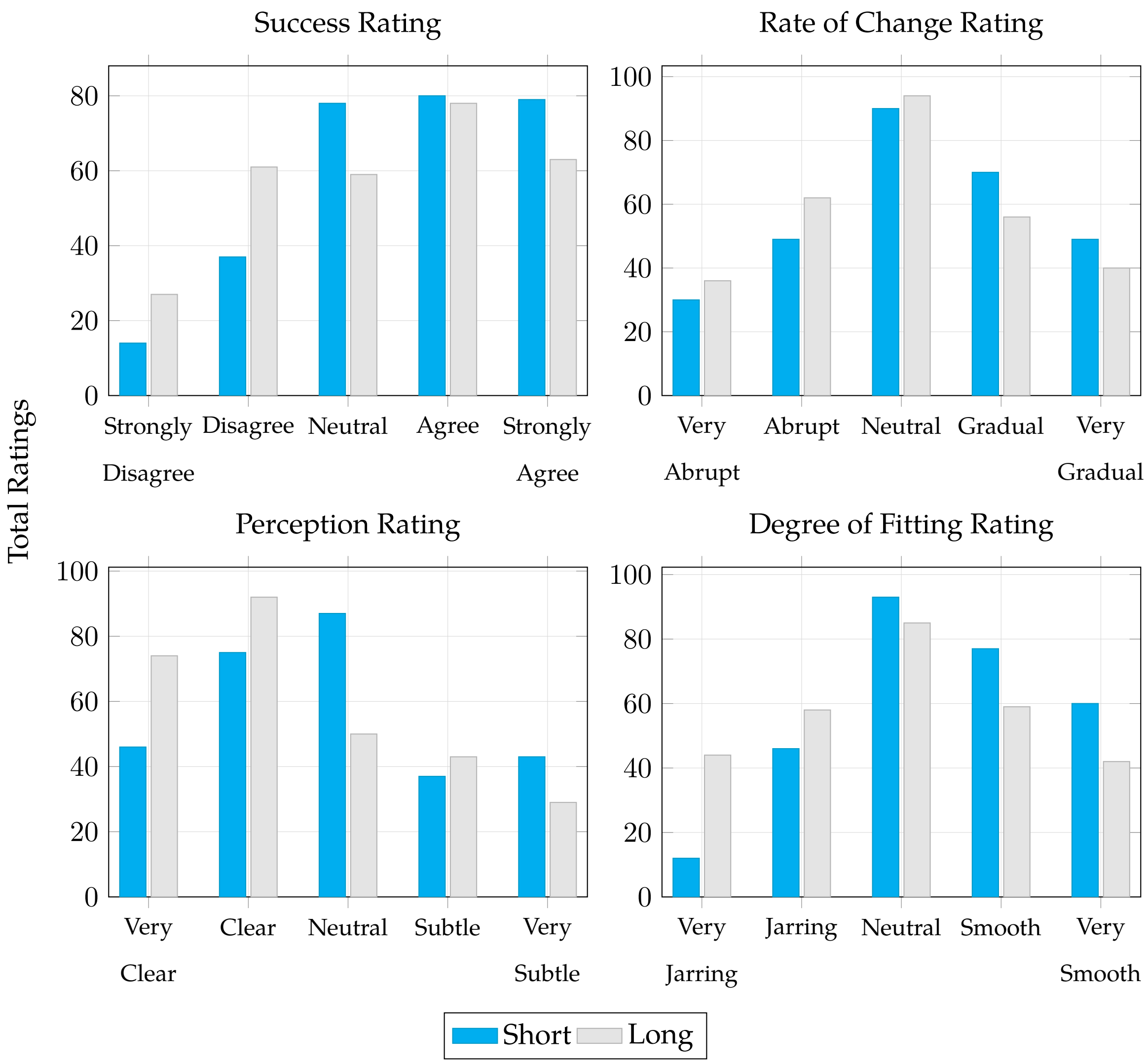


Figure 4.2: Ratings between transition lengths

A statistically significant difference was obtained between transition positions for ratings of degree of fitting, meaning that the null hypothesis is rejected. As illustrated in Fig. 4.3, outside-of-phrase transitions tend to be rated as smoother than within-phrase transition. This is presumably because a transition that occurs after a phrase has completed is musically less jarring, as any musical thought would have been completed during the phrase.

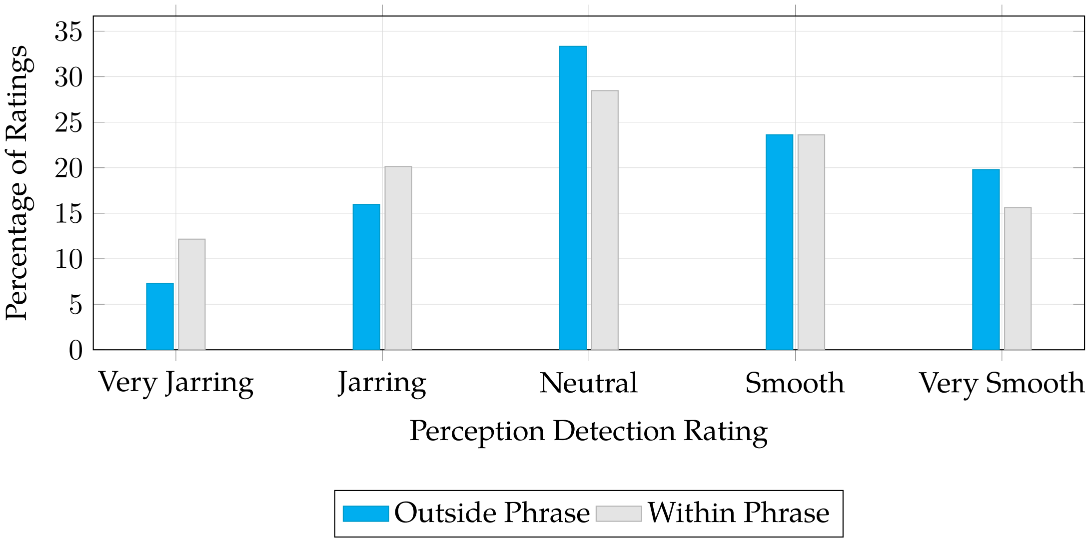


Figure 4.3: Degree of Fitting Ratings Between Transition Positions

The Wilcoxon Rank Sum Test (also known as the Mann-Whitney U Test) was used to compare the differences between means of two participant groups, with the null hypothesis being that there is no difference in ratings between the groups. In this case, the test was run 3 times based on participant gender, knowledge of harmony, and knowing how to play an instrument. The results are shown below in Table 4.6.

	Participant Gender, p	**Knowledge of Harmony, p**	**Instrument Knowledge, p**
Success	0.755	0.941	0.175
Rate of Change	0.79	0.975	0.77
Perception	**0.0269**	0.0785	0.544
Degree of Fitting	0.753	0.791	0.286

Table 4.6: Results of the Wilcoxon Rank Sum Test for participant gender, knowledge of harmony, instrument knowledge, and transition position

While the null hypothesis is not rejected for the majority of conditions (suggesting that there is no major difference between both populations for these pair of groups), a statistically significant difference was shown between participant genders for perception ratings. As illustrated in Fig. 4.4, women tend to rate transitions as being clearer when their results are compared to that of men.

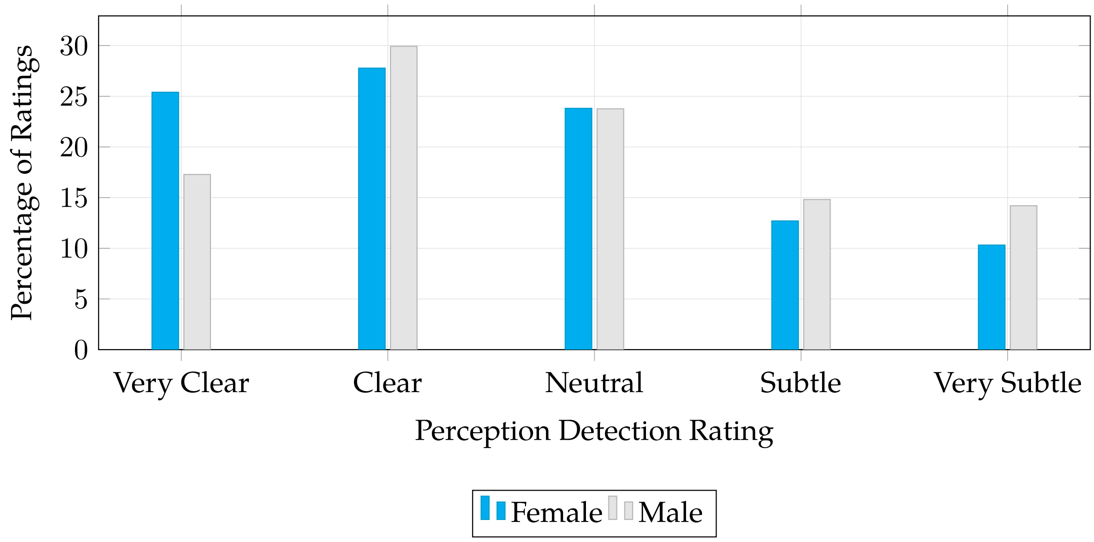


Figure 4.4: Perception Ratings Between Participant Gender

4.3 Observations

4.3.1 Transition Detections

As discussed in Chapter 3, the entire dataset was covered twice by a total of 36 participants. Fig. 4.5 illustrates one transition from the study with results from two participants. The colours of each line have the following meanings:

- **gray:** bar lines
- **green:** phrase markings (both beginning and end)
- **orange:** the actual start and end time of the transition
- **black:** transition detections by participants, with the top and bottom half of the figure corresponding to the different participants

- **blue:** 450ms after the transition has begun

- **pink:** 2000ms after the transition has begun

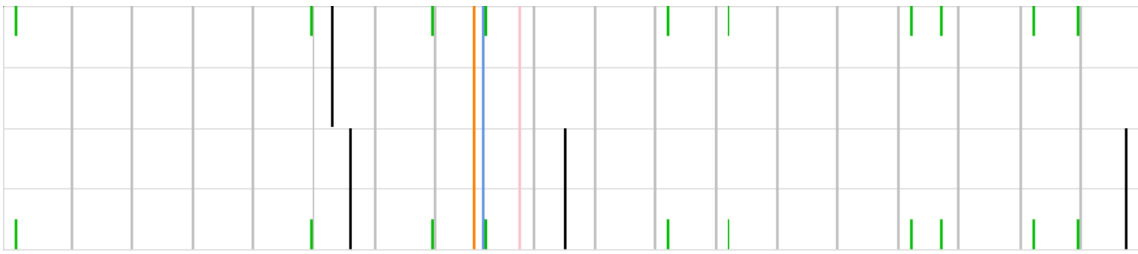


Figure 4.5: An example of a detected transition

For each study piece, the timing of the first detection made by participants after the actual start time of the transition was saved in order to determine how long it took participants to detect when the transition actually took place.

This result is illustrated in Fig. 4.6, which shows the percentage of all successfully detected transitions over time. This shows that there seems to be a significant increase in the number of transition detections between 1 and 2 seconds after the actual transition takes place. The size of this increase gradually decreases over time until it tapers off at around 7 seconds.

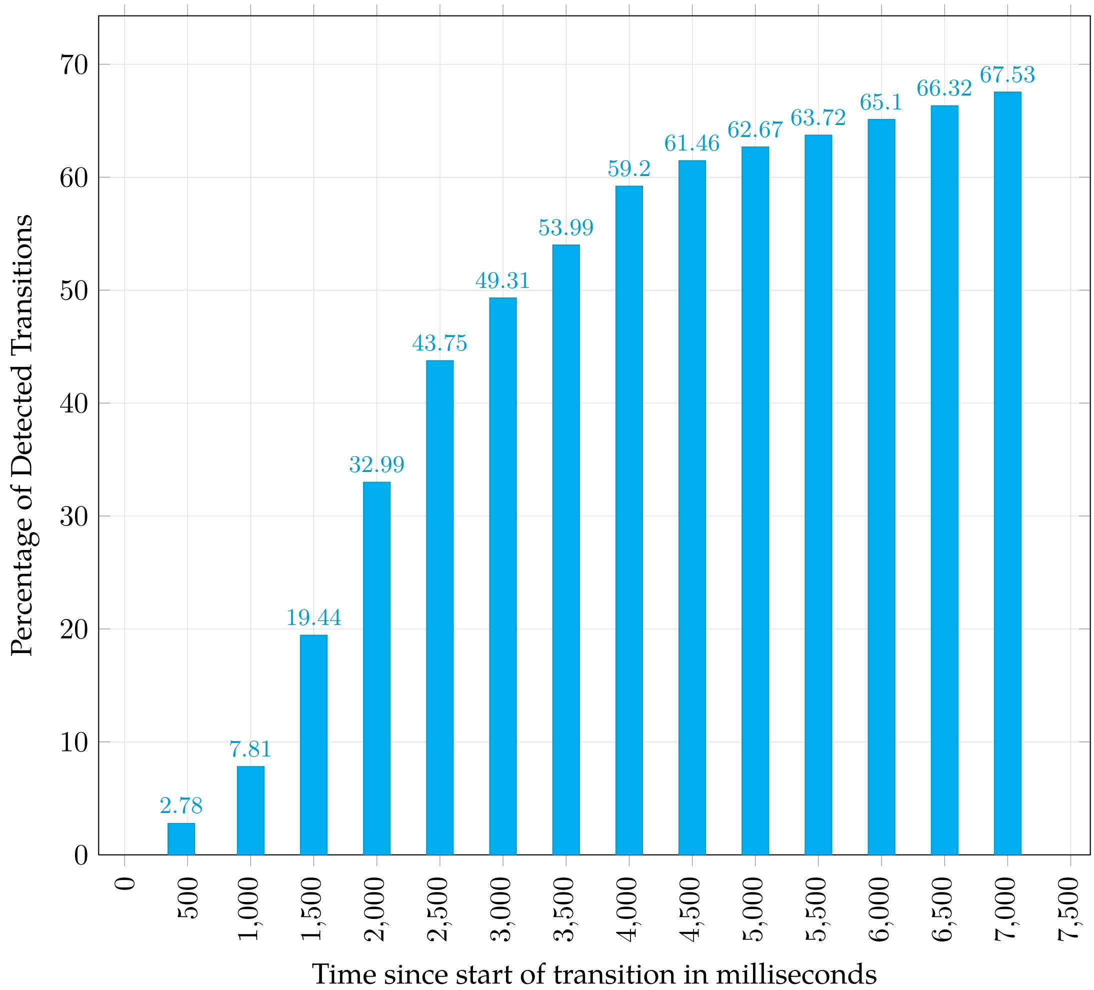


Figure 4.6: Percentage of all successfully detected transitions over time

Fig. 4.7 shows the percentage of successfully detected transitions over time for each transition technique used in the study pieces. Here, a significant number of both abrupt cut transitions and horizontal resequencing transitions seem to have been detected 1.5 seconds after the actual transition takes place. In contrast, both weighted averaging and crossfading transitions are detected much later, with almost 50% of crossfading transitions not being detected after 7 seconds.

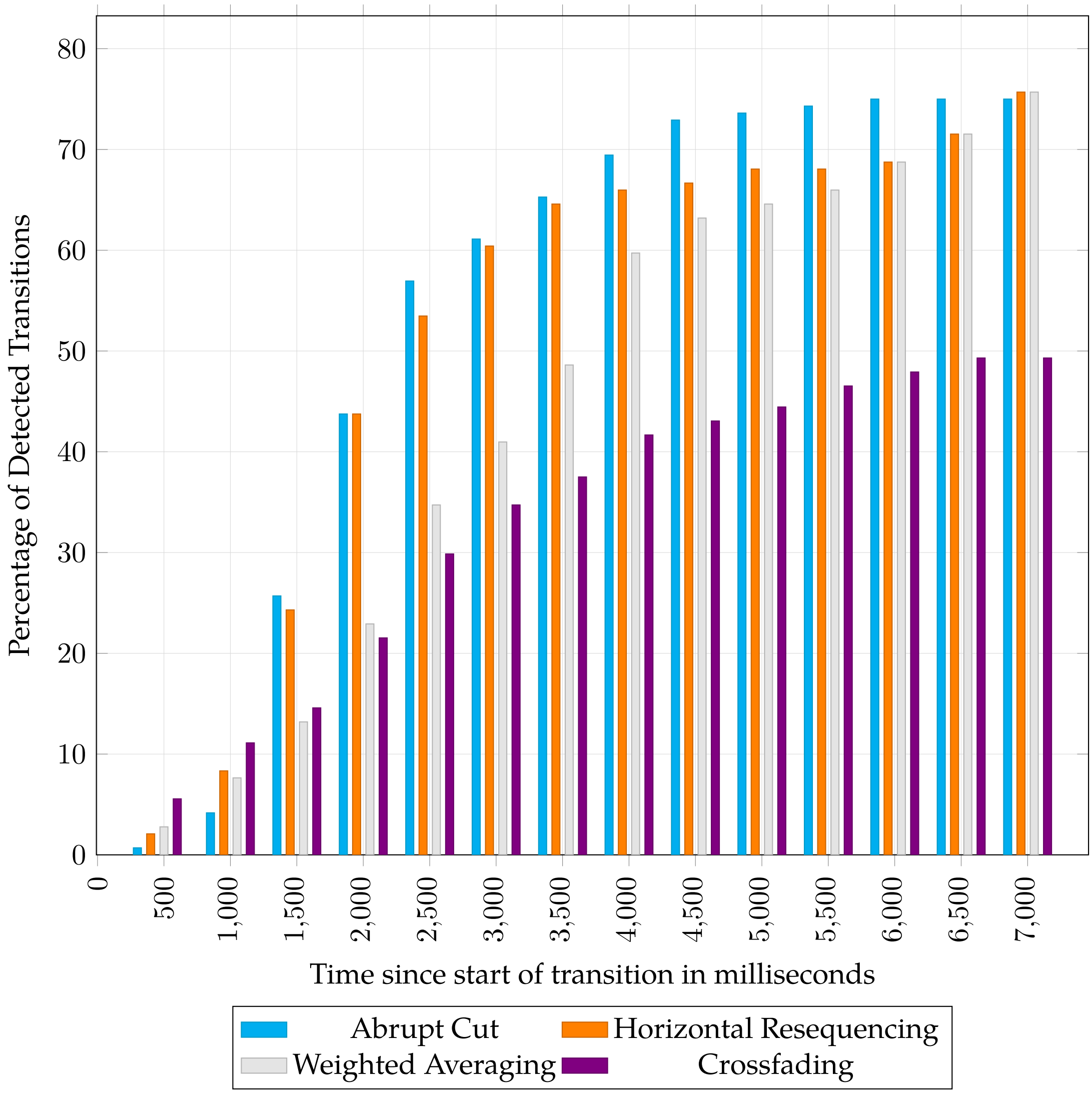


Figure 4.7: Percentage of Successfully Detected Transitions Over Time

4.3.2 Differences Between Transition Techniques

The Kruskal-Wallis rank sum test was used in order to determine if there were statistically significant differences between the transition techniques for each variable. The Dunn's test using the Benjamini–Hochberg correction was used as a post-hoc test in order to see which techniques contained significant differences.

4.3.2.1 Success Rating

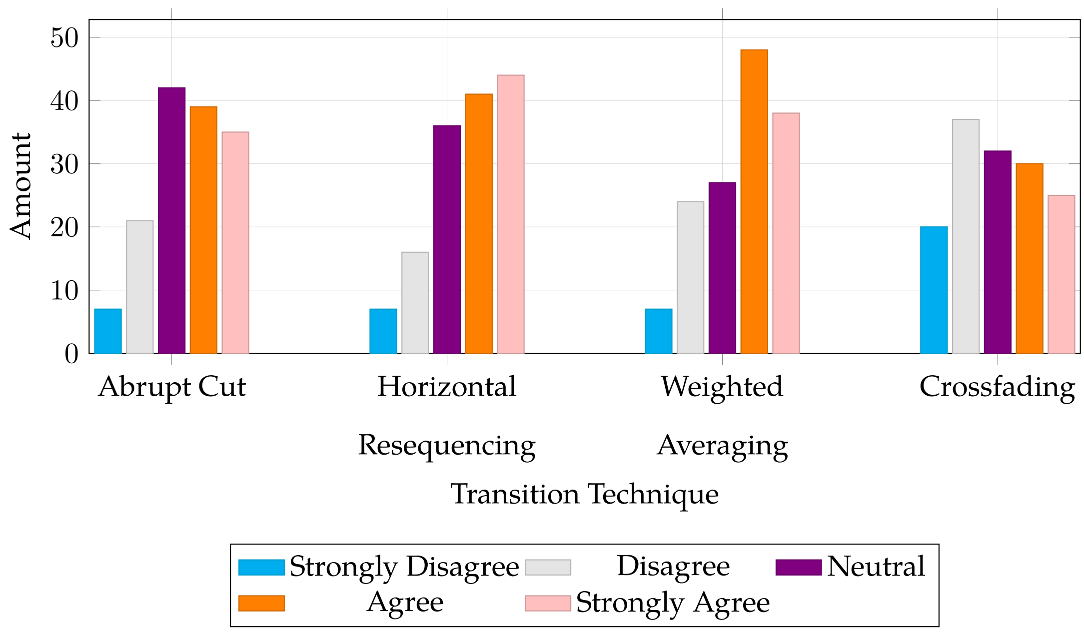


Figure 4.8: Success Ratings for each transition technique

The success ratings for all 4 transition techniques are illustrated in Fig. 4.8. The Kruskal-Wallis test showed that there was a statistically significant difference between all 4 transition techniques, $\chi^2 = 17.522$, df $= 3$, $p = 0.000552$. The Dunn's test showed that the statistically significant differences were between the crossfading technique and all other transition techniques, as shown below in Table 4.7.

First Technique	**Second Technique**	p**, adjusted**
Crossfade	Abrupt Cut	**0.00887**
Crossfade	Horizontal Resequencing	**0.00614**
Crossfade	Weighted Averaging	**0.000484**
Abrupt Cut	Horizontal Resequencing	0.991
Abrupt Cut	Weighted Averaging	0.399
Horizontal Resequencing	Weighted Averaging	0.489

Table 4.7: Results of the Dunn Test for success ratings between transition techniques

Overall, these results indicate that the crossfading technique seems to have been rated as less successful than the rest of the transition techniques within the context of this study.

4.3.2.2 Rate of Change Rating

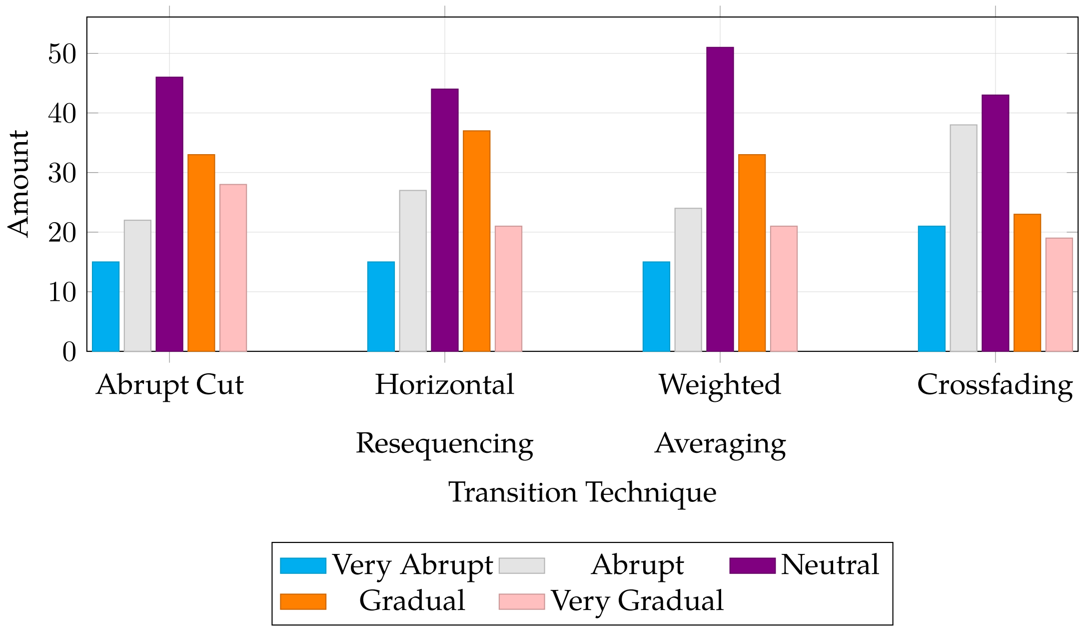


Figure 4.9: Rate of Change Ratings for each transition technique

The ratings for rate of change for all 4 transition techniques are illustrated in Fig. 4.9. The Kruskal-Wallis test showed that there was no statistically significant different between the transition techniques, $\chi^2 = 6.993$, df $= 3$, $p = 0.0721$.

4.3.2.3 Perception Rating

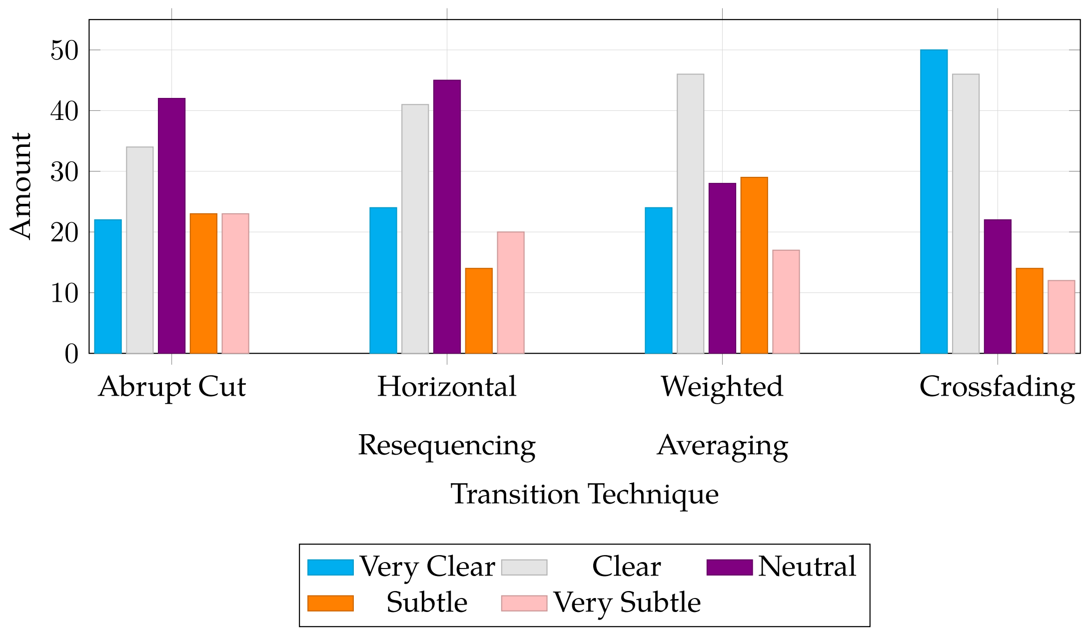


Figure 4.10: Perception Ratings for each transition technique

The perception ratings for all 4 transition techniques are illustrated in Fig. 4.10. The Kruskal-Wallis test showed that there was a statistically significant difference between all 4 transition techniques, $\chi^2 = 9.456$, df $= 3$, $p = 0.0238$. The Dunn's test showed that the statistically significant differences were between the crossfading technique and both the abrupt cut and weighted averaging techniques, as shown below in Table 4.8.

First Technique	Second Technique	p, adjusted
Crossfade	Abrupt Cut	**0.0427**
Crossfade	Horizontal Resequencing	0.328
Crossfade	Weighted Averaging	**0.03**
Abrupt Cut	Horizontal Resequencing	0.29
Abrupt Cut	Weighted Averaging	0.907
Horizontal Resequencing	Weighted Averaging	0.284

Table 4.8: Results of the Dunn Test for perception ratings between transition techniques

Overall, these results indicate that the crossfading technique seems to have been rated as clearer to perceive than both the abrupt cut transition and weighted averaging techniques within the context of this study.

This result is particularly surprising given the previous results. For example, as described in Section 4.3.1, the crossfading technique was detected later than all other transition techniques tested. This would imply that crossfading was therefore more difficult to perceive than the other transition techniques, which contradicts the results found in Table 4.8.

One possible interpretation as to why crossfading was detected later than the rest of the transitions may be because participants only counted the transition as having taken place when the crossfading had finished, rather than when it began.

4.3.2.4 Degree of Fitting

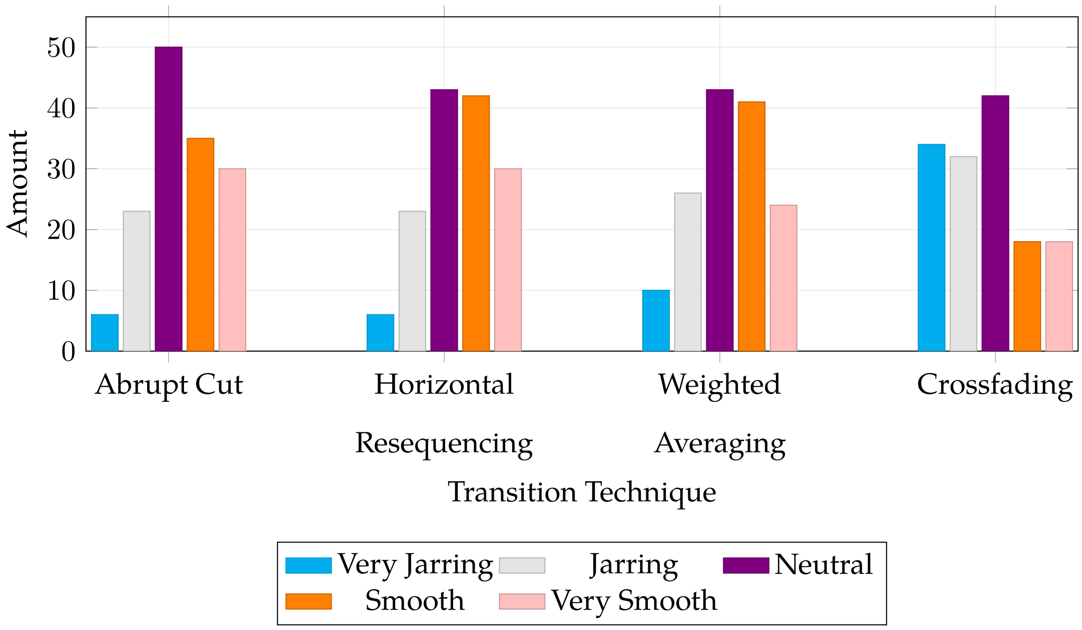


Figure 4.11: Degree of Fitting Ratings for each transition technique

The ratings for degree of fitting for all 4 transition techniques are illustrated in Fig. 4.11. The Kruskal-Wallis test showed that there was a statistically significant difference between all 4 transition techniques, $\chi^2 = 23.177$, df $= 3$, $p = 3.71 \times 10^{-5}$. The Dunn's test showed that the statistically significant differences were between the crossfading technique and all other transition techniques, as shown below in Table 4.7.

First Technique	**Second Technique**	p, **adjusted**
Crossfade	Abrupt Cut	**0.000381**
Crossfade	Horizontal Resequencing	**0.000409**
Crossfade	Weighted Averaging	**0.000171**
Abrupt Cut	Horizontal Resequencing	0.906
Abrupt Cut	Weighted Averaging	0.869
Horizontal Resequencing	Weighted Averaging	0.956

Table 4.9: Results of the Dunn Test for degree of fitting ratings between transition techniques

Overall, these results indicate that the crossfading technique seems to have been rated as more jarring than the rest of the transition techniques within the context of this study.

4.3.3 Comparison of Source and Target Pieces

From the data set of 288 pieces, 48 of these consisted of transitions between the same source and target pieces (for example: transitioning from the *Radetzky March* into the *Radetzky March* again). Since the data set was seen twice, this meant that 96 data points were from pieces that had the same source and target, while 480 data points were from pieces that have different source and targets (for a total of 576 data points).

The Wilcoxon Rank Sum Test was used to compared the differences between these two categories, with the null hypothesis being that there is no difference in ratings between them. The results can be seen in Table 4.10.

Category	**W**	p **value**
Success	16542	$p = 7.163 \times 10^{-6}$
Rate of Change	14967	$p = 2.405 \times 10^{-8}$
Perception	33468	$p = 6.324 \times 10^{-13}$
Degree of Fitting	14702	$p = 8.312 \times 10^{-9}$

Table 4.10: Results of the Wilcoxon Rank Sum Test when comparing transitions between source and target pieces

Figure 4.12: Ratings between pieces with the same source and target piece, and pieces with a different source and target piece

Thus, the null hypothesis is rejected, and the difference between the two groups is statistically significant, meaning that the more similar the source and target pieces are, the greater the possibility of the transition being ranked as successful, gradual, subtle, or smooth. This is illustrated in Fig. 4.12.

4.4 Conclusions

In conclusion, this study has shown that for the context presented in this particular study, there are associations between success, rate of change, degree of fitting, and perception of transitions. Transitions that are rated as being smooth tend to change gradually, smoothly, and subtly between the source and target pieces. Moreover, transitions that occur in between musical phrases tend to be rated as less jarring, and transitions that occur between the same source and target piece also tend to be rated as more successful. This implies that in order to create musical transitions for video games, following these criteria may lead to a more immersive experience. However, there is nothing stopping a composer or audio programmer from using these results to create transitions that are more jarring as a conscious artistic decision, as discussed in Section 3.1.

An interesting result is the tension between the amount of time needed to detect a crossfading transition compared with the ratings of all 4 criteria for this transition technique. While a crossfading transition tends to be detected fairly late by participants, they were also more likely to be rated as unsuccessful, very abrupt in terms of their rate of change, very clear in terms of their perception, and very jarring in terms of their degree of fitting between both source and target pieces. One possible interpretation is that while participants might not have detected the exact event at which the transition took place, nevertheless the accumulated effect over time must have been substantial enough for the transition to be deemed unsuccessful.

In a similar manner, Sweet (2014, p. 167) states that longer transitions are considered to be more effective. The results obtained from this study contradict this, as shorter transitions (such as abrupt cut transition and horizontal resequencing) were rated as more successful, smoother, subtler, and more gradual than longer transitions. However, both longer and shorter transitions may be considered as different tools in a game composer's toolbox and may be used when appropriate in a variety of different scenarios. While the results obtained in this study show that shorter transitions were preferred over longer transitions in this context, this does not necessarily mean that shorter transition techniques should be used in all situations. In particular, a composer should consider the different pieces of music being used, and that the effectiveness of longer transitions may depend on the similarity between the two pieces.

Finally, while the weighted averaging technique did not exceed any one particular transition technique in terms of success, rate of change, perception, and degree of fitting ratings, there are several improvements that can be applied to this technique, as will be discussed in Chapter 5.

Supposing, for instance, that the fundamental relations of pitched sounds in the science of harmony and of musical composition were susceptible of such expression and adaptations, the engine might compose elaborate and scientific pieces of music of any degree of complexity or extent.

Ada Lovelace, translator's note in Luigi Menabrea's "Sketch of the Analytical Engine invented by Charles Babbage", p. 694[1]

5 Towards a New Transition Algorithm

This chapter describes the technical details behind the development of a new transition algorithm to be used in a video game context. Section 5.1 investigates how transitions are normally implemented in video games, and introduces the concept of a transition region. Section 5.2 introduces the multiple viewpoint system by discussing viewpoints and the use of Markov models within such a system. Finally, Section 5.3 discusses the iterative approach used to find a suitable algorithm for generating transitions using a multiple viewpoint system within the context of a video game.

[4]Luigi Federico Menabrea (1843). "Sketch of the Analytical Engine invented by Charles Babbage". In: *Scientific Memoirs: Selected from the Transactions of Foreign Academies of science and Learned Societies and from Foreign Journals*. Ed. by Richard Taylor. Trans. and comm. by Ada Lovelace. Vol. 3. London, United Kingdom: Richard and John E. Taylor. Chap. 29, pp. 666–731

5.1 Rethinking Transitions

As discussed in Chapter 2 Section 2.3.3, transitions in games can be placed in one of two categories: zone transitions, and event-triggered transitions. This section will discuss the development of an algorithm used to algorithmically generate suitable musical transitions between different zones.

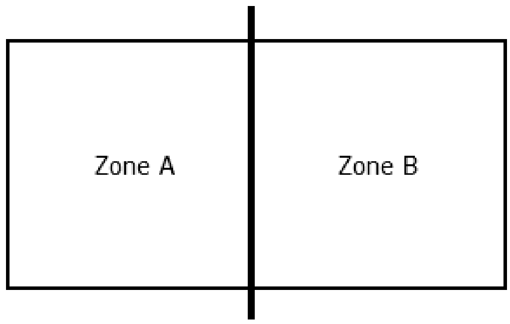


Figure 5.1: Boundary line between between zones A and B

In Chapter 2 Section 2.3.3, some common transition techniques used in video games were discussed, such as horizontal resequencing, vertical reorchestration, and crossfading. Any of these transitions may be used when transitioning between different zones. A common implementation for each of these techniques is for an event to trigger when crossing a zone boundary, as can be seen in Fig. 5.1. This event lets the game know that the player's position has moved across the boundary from the first to the second zone, and that the music should change accordingly. However, this requires an instantaneous musical change between the source piece and the target piece.

As described earlier in Section 2.3.4, a musical transition that must happen immediately after it has been triggered will either sound jarring, or happen after some amount of delay. This issue is further complicated due to the player having direct control over the movement over their character in the virtual environment, meaning that unlike in a linear scenario such as a film, transitions cannot be prepared beforehand, since it is unknown as to when they will actually take place.

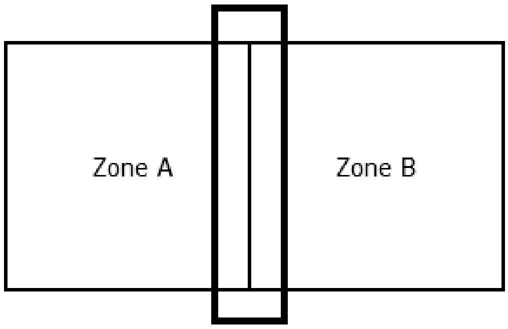


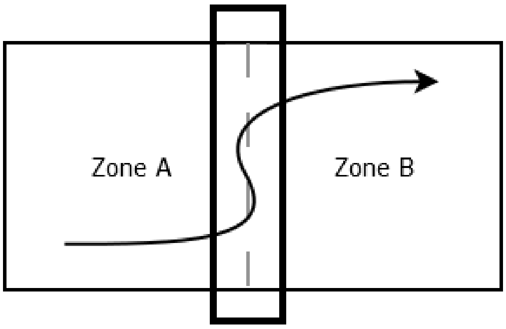
Figure 5.2: Transition region between zones A and B

One solution to this issue is to have a transition region that plays a generated transition between the two pieces, as illustrated in Fig. 5.2. This region serves as a buffer zone, and the major difference that this has over the use of a zone boundary is that it may have a variable width[2] which may be set individually for each transition region used. In contrast, as stated before, a zone boundary has no width, meaning that musical transitions must occur immediately after crossing the threshold. Having a variable width allows transition regions to be as wide as necessary to allow the music to transition accordingly into the new piece representing the new zone.

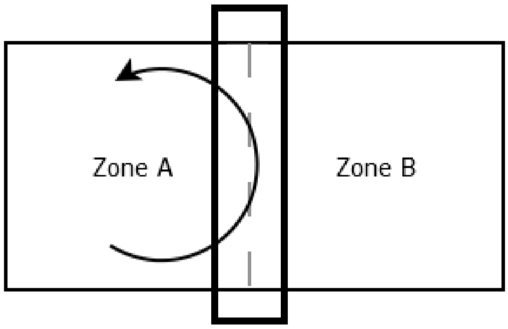
However, player interaction must still be taken into account when considering the use of a transition region. Fig. 5.3 demonstrates various different player behaviours that must be accounted for when considering musical transitions between zones, and the system should be able to cater for any of these behaviours.[3]

[2]Different game engines tend to work with different units of measurement, so a precise value tends to be specific to the implementation of the transition region.

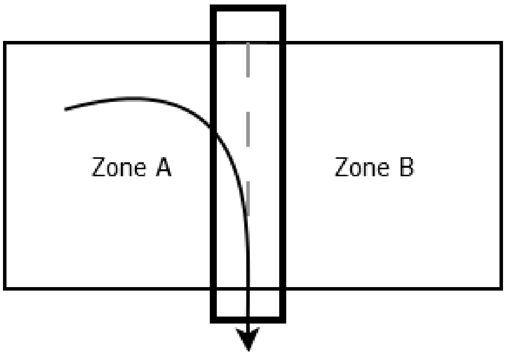
[3]While Fig. 5.3 describes theoretical player behaviour that must be accounted for when implementing transition regions, the study described in Chapter 6 only takes into account the behaviour shown in Figs. 5.3a and 5.3d as can be seen in Section 6.2.2.



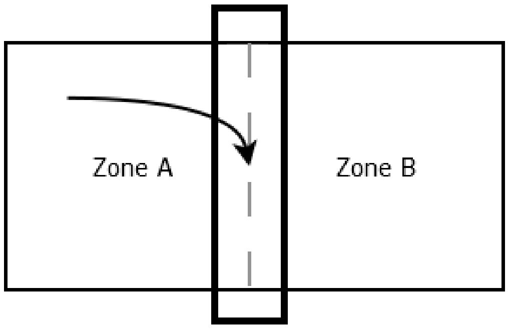
(a) Seamless transition between A and B



(b) Starting at A, entering the transition region, and leaving the transition region back to A



(c) Starting at A, entering the transition region, and remaining inside it



(d) Starting at A, entering the transition region, and stopping inside it

Figure 5.3: Examples of player movement while moving between two zones

5.2 Building A New Transition Algorithm

Several algorithmic techniques used in computer music were discussed in Section 2.2.2. Markov models were chosen due to their suitability when working with melodies and their effectiveness when working with style imitation, both of which are relevant when generating transitions between two melodies. The following section further details the use of Markov models in order to generate music, and introduces the multiple viewpoint system as the technique that will be used in this dissertation.

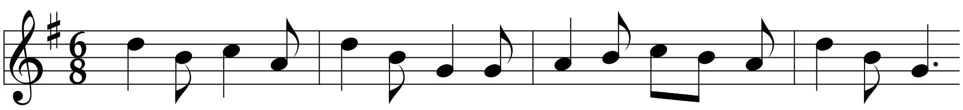
5.2.1 Markov models

According to Nierhaus (2008, p. 68), a Markov chain can be defined as a form of stochastic process which has the following features:

- As a stochastic process, a Markov chain is able to model a sequence of events, where a unique event at time t is represented by e_t. Each unique event in the sequence is represented as a state in the chain.

- The probability of a state is dependent on the previous n states (referred to by Whorley (2013, p. 50) as the *context*) that have appeared before it. A first-order Markov chain therefore generates the probability of the next state based on a single previous state, while a second-order Markov chain generates the probability of the next state based on the previous two states, and so on. Markov chains can therefore be referred to as n-gram models, where n is the size of the chain's context.

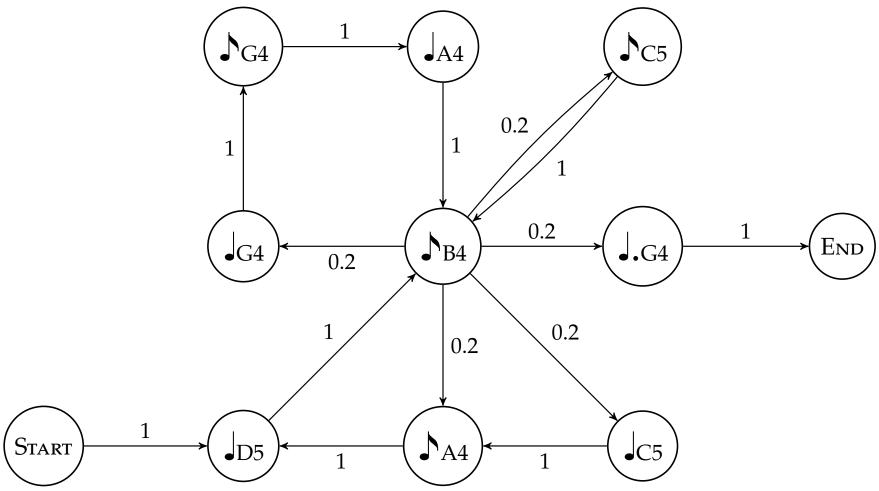
A Markov chain can be represented as a transition matrix of probabilities, or as a directed graph with nodes representing events and edges representing the probability of predicting one event given another. Example 5.1 shows the first four bars of the melody in the nursery rhyme *Girls and Boys*, taken from Rimbault (1846, p. 5). Fig. 5.4 illustrates the representations of a first-order Markov chain made from said example, with Fig. 5.4a illustrating the transition matrix of probabilities, and Fig. 5.4b illustrating the directed graph.



Example 5.1: The first two bars of the nursery rhyme *Girls and Boys* from Rimbault (1846, p. 5)

	♪G4	♩G4	♩.G4	♪A4	♩A4	♪B4	♪C5	♩C5	♩D5	End
♪G4	0	0	0	0	**1**	0	0	0	0	0
♩G4	**1**	0	0	0	0	0	0	0	0	0
♩.G4	0	0	0	0	0	0	0	0	0	**1**
♪A4	0	0	0	0	0	0	0	0	**1**	0
♩A4	0	0	0	0	0	**1**	0	0	0	0
♪B4	0	**0.2**	**0.2**	**0.2**	0	0	**0.2**	**0.2**	0	0
♪C5	0	0	0	0	0	**1**	0	0	0	0
♩C5	0	0	0	**1**	0	0	0	0	0	0
♩D5	0	0	0	0	0	**1**	0	0	0	0

(a) Transition matrix for the first two bars of *Girls and Boys*



(b) Transition chain for the first two bars of *Girls and Boys*

Figure 5.4: Representations of a first-order Markov chain made from Example 5.1

The transition matrix of probabilities in Fig. 5.4a represents every probability of transitioning from the one state (as seen on each row) to another (as seen in every column). As stated by Ames (1989), since each number in a cell represents a probability, by definition a row of numbers must add up to 1 (or an end state has been reached). A transition chain, as seen in Fig. 5.4b represented by a directed graph, is a graphical representation of the transitions between states, and is used to represent the model's transition matrix and the probability of transitioning from state to state.

While Fernández and Vico (2014) discuss the use of different types of Markov models such as hidden Markov models and Markov decision processes, this dissertation will be using Markov chains as discussed above.

5.2.2 Viewpoints

This dissertation makes use of viewpoints in order to model abstract attributes of a musical event from a sequence of events, rather than modelling the entire event itself. In essence, a viewpoint is simply a lens that allows a sequence of musical events to be converted into a sequence of attributes taken from those events. By using several viewpoints inside a multiple viewpoint system and combining them accordingly, a distributed approach can be used to model music (Conklin 1990).

A melody can be represented as a sequence of individual musical events e_1^i, where i is the length of the sequence.[4] Each event e is made up of a collection of basic attributes that describe abstract properties of such events, and all attributes (whether basic or otherwise) are represented by their type τ. Thus, each event can be formally defined as an n-tuple of basic types (M. Pearce 2005), as can be seen in Equation 5.1.[5]

$$e\colon \{\tau_{b_1}, \tau_{b_2}, \ldots, \tau_{b_n}\} \tag{5.1}$$

The domain $[\tau]$ of an event type τ defines the set of all possible values that the type may contain. Normally, these are numeric values that may be mapped onto the attributes of a particular event. The semantic domain of a type, represented by $[\![\tau]\!]$, is described by T. Hedges (2017, p. 63) as being the set of domain values $[\tau]$ represented using values that are easier for people to conceptually understand. A mapping from a domain value to a semantic domain value is given by $[\![\cdot]\!]_\tau\colon [\tau] \rightarrow [\![\tau]\!]$. Similarly, a mapping from a semantic

[4]While this approach has been extended to harmony, notably by Whorley (2013) and T. Hedges (2017), the rest of this dissertation will be framed with reference to melody.

[5]Note that most equations have been taken from T. Hedges (2017), unless as otherwise stated here.

domain value to a domain value is done via the following function: $[\![\cdot]\!]\prime_\tau : [\![\tau]\!] \rightarrow [\tau]$. An example of this can be seen in Table 5.1 below.

The set of all possible events is denoted by ξ, where $e \in \xi$. This is made up of the product of domains of any relevant basic types, as can be seen in Equation 5.2.

$$\xi = [\tau_{b_1}] \times [\tau_{b_2}] \times \cdots \times [\tau_{b_N}] \tag{5.2}$$

Formally, a viewpoint is defined by the partial function Ψ_τ as shown in Equation 5.3, which allows a sequence of events in ξ^* to be mapped onto the viewpoint's type τ. ξ^* represents the Kleene closure of ξ which contains all possible sequences of events, including the empty sequence. This function is typically applied to each event in a sequence, iterating over the complete sequence, in order to convert a sequence of notes to a sequence of viewpoint values. For example, a list of notes could be mapped to a list of its corresponding pitch values.

$$\Psi_\tau : \xi^* \rightharpoonup [\tau] \tag{5.3}$$

In Conklin (1990, p. 59), a viewpoint is formally defined as consisting of both the viewpoint function Ψ_τ and the associated model. The approach used in this dissertation is one similar to M. Pearce (2005) and T. Hedges (2017), where the statistical model is separated from the definition of the viewpoint. This is to allow for a multitude of different models associated with the same matching function, and allows for statistical models to be made based on the corresponding sequence of types, rather than just the entire event itself. As stated by T. Hedges (2017, p. 66), while any finite context model may be used to model a viewpoint, the vast majority of the literature has used Markov models, and this dissertation will do the same.

Each viewpoint also contains a type set, as shown below in Equation 5.4, which as stated by Conklin and Witten (1995), shows which types the viewpoint is made up from and therefore which types the viewpoint can predict. In the case of basic viewpoints, the type set is simply the viewpoint type itself.

$$\langle\tau\rangle \subseteq \{\tau_{b_1}, \ldots, \tau_{b_n}\} \tag{5.4}$$

A summary of the notation and functions related to typed attributes is given below in Table 5.1.

Symbol	Interpretation	Example
τ	A typed attribute	`cpitch`
$[\tau]$	Syntactic domain of τ	$\{60, \ldots, 72\}$
$\langle\tau\rangle$	Type set of τ	$\{$`cpitch`$\}$
$[\![\tau]\!]$	Semantic domain of τ	$\{\{\mathrm{B\#}_3, \mathrm{C}_4, \ldots\}, \ldots, \{\mathrm{B\#}_4, \mathrm{C}_5, \ldots\}\}$
$[\![\cdot]\!]_\tau : [\tau] \to [\![\tau]\!]$	Semantic interpretation of $[\tau]$	$[\![60]\!] = \{\mathrm{B\#}_3, \mathrm{C}_4, \ldots\}$
$[\![\cdot]\!]'_\tau : [\![\tau]\!] \to [\tau]$	Syntactic interpretation of $[\![\tau]\!]$	$[\![\mathrm{B\#}_3]\!]' = 60$
$\Psi_\tau : \xi^* \rightharpoonup [\tau]$	Viewpoint function	Projection function

Table 5.1: Sets and functions associated with typed attributes, replicated from T. Hedges (2017, p. 64)

While a matching function is introduced by Conklin (1990, pp. 59–60) in order to remove undefined values from sequences, as can be seen in Equation 5.5, it is not used in this dissertation in order to preserve those null values, since they may represent rests.

$$\Phi_\tau : \xi^* \rightharpoonup [\tau]^* \tag{5.5}$$

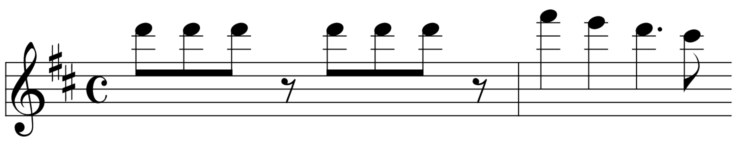
5.2.3 Viewpoint Classes

In the literature, several viewpoints are grouped in different classes, with each class able to model different sequences in different ways. These classes are:

- basic viewpoints

- derived viewpoints
- linked viewpoints
- test viewpoints
- threaded viewpoints
- relational viewpoints

Each of the viewpoint classes are described in more detail below. The first two bars of Strauss's *Radetzky-Marsch, zu Ehren des grossen Feldherrn* (Strauss n.d.) is used to illustrate each viewpoint type, and the values for each chosen viewpoint are displayed in a solution array.

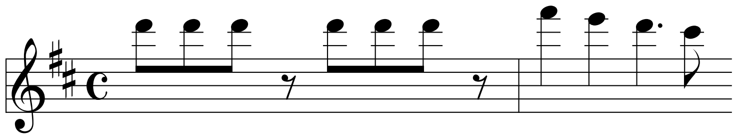


Position	1	2	3	4	5	6	7	8	9	10	11	12
Music Event	8D6	8D6	8D6	8R	8D6	8D6	8D6	8R	4F#6	4E6	4.D6	8C#6

Figure 5.5: The first two bars of Strauss's *Radetzky March, Op. 228*, used to illustrate each viewpoint class

5.2.3.1 Basic Viewpoints

Basic viewpoints model basic types that make up the musical event. This means that a basic viewpoint Ψ_{τ_b} operates on a single element and simply returns the corresponding basic type. Hence, the type set $\langle\tau\rangle$ of a basic viewpoint is the basic type itself. Four basic viewpoints that are used in this dissertation can be seen in Table 5.6, and are described in more detail below.



Position	1	2	3	4	5	6	7	8	9	10	11	12
Onset	0	12	24	36	48	60	72	84	96	120	144	180
Pitch	86	86	86	$\perp$	86	86	86	$\perp$	90	88	86	85
Duration	8	8	8	8	8	8	8	8	4	4	4.	8
Octave	6	6	6	$\perp$	6	6	6	$\perp$	6	6	6	6

Figure 5.6: Examples of basic viewpoints `Pitch` and `Duration`

`Onset`

`Onset` returns the event's onset in time from the beginning of the melody. In a similar manner to M. Pearce (2005, p. 63), a timebase value is used to determine the "granularity of the time representation" of an event's duration; here, the timebase is set to 96.[6]. The domain $[\tau]$ of the `Onset` viewpoint corresponds to $\mathbb{Z}^*$.[7] Musical events that do not contain pitch information (such as rests) will return undefined values, which are represented as $\perp$.

This viewpoint is not used in this dissertation on its own, but is used to create the derived viewpoint `IOI` (and any linked viewpoints that use that derived viewpoint).

`Pitch`

`Pitch` returns the MIDI note number that corresponds with the note's pitch and octave. If the event is a rest, an undefined value (represented as $\perp$) is returned. The domain $[\tau]$ corresponds to $\mathbb{Z}$, but is most often restricted to $\{36, \ldots, 96\}$, representing a range of pitches between the C found 2 octaves be-

[6]For example, the value of a crotchet (or quarter note) becomes 24, and the value of a quaver (or eighth note) becomes 12 A full list of each note duration and their corresponding value with a timebase of 96 can be seen in the Appendix F.

[7]In this context, $\mathbb{Z}^*$ is used to define non-negative integers.

low middle C and the C found 3 octaves above middle C. As a basic viewpoint, the type set $\langle\tau\rangle$ is therefore `Pitch` itself.

This viewpoint is used in this dissertation both on its own, as well as part of other derived viewpoints such as `Interval` and linked viewpoints such as `Pitch` $\otimes$ `Duration`.

`Duration`

`Duration` returns the duration value associated with the musical event. The domain $[\tau]$ corresponds to $\mathbb{Z}^+$, but is most often restricted to the values $\{3, \ldots, 96\}$ representing a range of durations between a demisemiquaver (or 32nd note) and a semibreve. These values assume a timebase value of `96`, as discussed earlier. As a basic viewpoint, the type set $\langle\tau\rangle$ is therefore `Duration` itself.

This viewpoint is used in this dissertation both on its own, as well as part of other derived viewpoints such as `DurationRation` and linked viewpoints such as `Pitch` $\otimes$ `Duration`.

`Octave`

`Octave` returns the octave value associated with the note. If the event is a rest, an undefined value is returned. The domain $[\tau]$ corresponds to $\mathbb{Z}$, but a range of $\{2, \ldots, 7\}$ is commonly used.

This viewpoint is used in this dissertation both on its own, as well as part of linked viewpoints such as `Octave` $\otimes$ `DurationRatio`.

The basic types `Pitch` and `Duration` will be used to define the event space; i.e. the basic types needed to create an event, as described in Equation 5.2, such that $e = \{\texttt{Pitch}, \texttt{Duration}\}$.

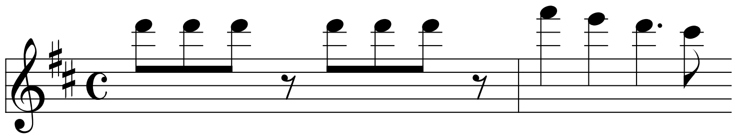
5.2.3.2 Derived Viewpoints

Derived viewpoints are made up of one or more basic viewpoints and can model relationships between adjacent events in a sequence. In practice, this is done by iterating over the sequence of events and applying the viewpoint function Ψ to two distinct elements at a time. A derived viewpoint may contain undefined values (denoted as $\perp$), since a viewpoint function is partial (M. Pearce 2005). The type set $\langle\tau\rangle$ of a derived viewpoint is the set of all basic types that make up the viewpoint.

$$\Psi_{\tau_d}(e^j) = \begin{cases} \perp & \text{if } j = 1 \\ (\Psi_{\tau_b}(e^j) - \Psi_{\tau_b}(e^{j-1}) & \text{otherwise.} \end{cases} \tag{5.6}$$

Previous implementations of derived viewpoints represented them as a piecewise function as seen in Equation 5.6, which returned a value based on a comparison between two adjacent events in a sequence. However, in this dissertation, since basic viewpoints may contain undefined values (such as rests not having a `Pitch` value), a comparison is made by skipping any undefined values until a defined value is reached. If no defined value is reached, then the function returns $\perp$.

Five derived viewpoints used in this dissertation can be seen in Table 5.7, and are described in more detail below to serve as a better example of how derived viewpoints work.



Position	1	2	3	4	5	6	7	8	9	10	11	12
Contour	⊥	0	0	⊥	0	0	0	⊥	1	-1	-1	-1
Interval	⊥	0	0	⊥	0	0	0	⊥	4	-2	-2	-1
DurationRatio	⊥	1	1	1	1	1	1	1	2	1	1.5	$0.\dot{3}$
ScaleDegree	0	0	0	⊥	0	0	0	⊥	4	2	0	11
IOI	⊥	12	12	12	12	12	12	12	12	24	24	36

Figure 5.7: Examples of derived viewpoints `Interval` and `Duration`

`Contour`

`Contour` is a derived viewpoint, and returns an indication of whether the current note has a higher or lower pitch than the previous note. The domain $[\tau]$ of this viewpoint is $\{\perp, -1, 0, 1\}$, as can be seen in Equation 5.7 below.

$$\Psi_{\text{contour}}(e^j) = \begin{cases} \perp & \text{if } \Psi_{\text{pitch}}(e^j) \text{ is a rest} \\ -1 & \text{if } \Psi_{\text{pitch}}(e^j) < \Psi_{\text{pitch}}(e^{j-1}) \\ 0 & \text{if } \Psi_{\text{pitch}}(e^j) = \Psi_{\text{pitch}}(e^{j-1}) \\ 1 & \text{if } \Psi_{\text{pitch}}(e^j) > \Psi_{\text{pitch}}(e^{j-1}) \end{cases} \tag{5.7}$$

As described earlier in Section 5.2.3, the `Contour` viewpoint (and all other derived viewpoints used in this research) have been amended to cater for undefined values due to rests. If $\Psi_{\text{pitch}}(e_1^{j-1})$ is a rest, the previous musical event is checked. This process is repeated until a note is found, and $\perp$ is returned if no note has been found.

This viewpoint is not used in this dissertation on its own, but is used as part of linked viewpoints such as `Contour` $\otimes$ `Duration`.

Interval

`Interval` is a derived viewpoint, and returns the interval between two pitches. The domain $[\tau]$ of this viewpoint is $\mathbb{Z}$, but is most often restricted to $\{-24, \ldots, 24\}$ for practical reasons, as it allows a range of two intervals in either direction, while the type set $\langle\tau\rangle$ for this implementation is the basic viewpoint `Pitch`. The viewpoint function for `Interval` can be seen below in Equation 5.8.

$$\Psi_{\text{interval}}(e^j) = \begin{cases} \perp & \text{if } j = 1 \text{ or } \Psi_{\text{pitch}}(e^j) \text{ is a rest} \\ \Psi_{\text{pitch}}(e^j) - \Psi_{\text{pitch}}(e^{j-1}) & \text{otherwise.} \end{cases} \tag{5.8}$$

This viewpoint is both used in this dissertation on its own, as well as part of linked viewpoints such as `Interval` $\otimes$ `DurationRatio`.

DurationRatio

`DurationRatio` is a derived viewpoint, and returns the ratio of durations between two events. The domain $[\tau]$ of this viewpoint is $\mathbb{Q}$, while the type set $\langle\tau\rangle$ for this implementation is the basic viewpoint `Duration`. The viewpoint function for `DurationRatio` can be seen below in Equation 5.9.

$$\Psi_{\text{durationratio}}(e^j) = \begin{cases} \perp & \text{if } j = 1 \\ \frac{\Psi_{\text{duration}}(e^j)}{\Psi_{\text{duration}}(e^{j-1})} & \text{otherwise.} \end{cases} \tag{5.9}$$

This viewpoint is both used in this dissertation on its own, as well as part of linked viewpoints such as `Interval` $\otimes$ `DurationRatio`.

ScaleDegree

`ScaleDegree` is a derived viewpoint, and returns the interval from the tonic of the piece. The domain $[\tau]$ of this viewpoint is $\{0, \ldots, 11\}$, and its corresponding viewpoint function can be seen below in Equation 5.10.

$$\Psi_{\text{scaledegree}}(e^j) = \begin{cases} \perp & \text{if } j = 1 \text{ or } \Psi_{\text{pitch}}(e^j) \text{ is a rest} \\ \Psi_{\text{pitch}}(e^j) - \text{piece's tonic} & \text{otherwise.} \end{cases} \tag{5.10}$$

This viewpoint is used as part of linked viewpoints such as `ScaleDegree` $\otimes$ `DurationRatio`.

IOI

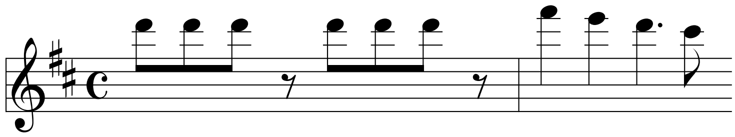
`IOI` is a derived viewpoint, and returns the inter-onset interval between two musical events. The domain $[\tau]$ of this viewpoint corresponds to $\mathbb{Z}^+$, while the type set $\langle\tau\rangle$ for this implementation is the basic viewpoint `Duration`. The relevant viewpoint function can be seen below in Equation 5.11.

$$\Psi_{\text{ioi}}(e^j) = \begin{cases} \perp & \text{if } j = 1 \\ \Psi_{\text{onset}}(e^j) - \Psi_{\text{onset}}(e^{j-1}) & \text{otherwise.} \end{cases} \tag{5.11}$$

This viewpoint is used as part of linked viewpoints such as `Pitch` $\otimes$ `IOI`.

5.2.3.3 Test Viewpoints

Test viewpoints model types with a Boolean value determined by the result of a test applied to each event in the sequence. Such tests may include whether or not a musical event is the first event in a bar. T. Hedges (2017) states that though these viewpoints can predict the basic types that they are derived from, they are more often used together with threaded viewpoints.

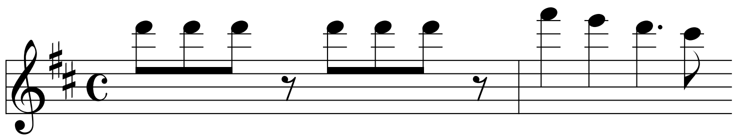


Position	1	2	3	4	5	6	7	8	9	10	11	12
`FirstInBar`	T	F	F	F	F	F	F	F	T	F	F	F

Figure 5.8: Example of test viewpoint `FirstInBar`

5.2.3.4 Threaded Viewpoints

Threaded viewpoints can model types whose values are only defined at certain points in the sequence. Here, M. Pearce (2005) states that these viewpoints were developed to take advantage of larger scale structures and patterns in music. Threaded viewpoints are made up of what T. Hedges (2017) calls base viewpoints (such as derived viewpoints or linked viewpoints) and test viewpoints, and are notated as $\tau_{\text{base}} \ominus \tau_{\text{test}}$.



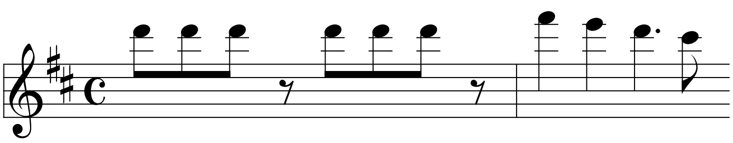
Position	1	2	3	4	5	6	7	8	9	10	11	12
`Interval` $\ominus$ `FirstInBar`	$\{T, 0\}$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\{T, 4\}$	$\perp$	$\perp$	$\perp$

Figure 5.9: Example of threaded viewpoint `Interval` $\ominus$ `FirstInBar`

5.2.3.5 Linked Viewpoints

Linked viewpoints, also represented as a piecewise function as seen in Equation 5.12, are made up of the product of what T. Hedges (2017) calls *primitive viewpoints*; any viewpoint that is not linked or threaded. Linked viewpoints are able to model interactions between individual types (Conklin and Witten 1995), and are defined by a product type τ_p over n constituent types, such that $\tau_p = \tau_1 \otimes \cdots \otimes \tau_n$. The domain of a linked viewpoint is therefore $[\tau_p] = [\tau_1] \times \cdots \times [\tau_n]$, and the type set of a linked viewpoint is $\langle\tau_p\rangle = \bigcup_{k=1}^{n} \langle\tau_k\rangle$. Two examples can be seen below in Table 5.10.

$$\Psi_\tau(e_1^j) = \begin{cases} \perp & \text{if } \Psi_{\tau_j}(e^j) = \perp \text{ for any } j \in 1,\ldots,n \\ (\Psi_{\tau_1}(e^j),\ldots,\Psi_{\tau_n}(e^j)) & \text{otherwise.} \end{cases} \tag{5.12}$$

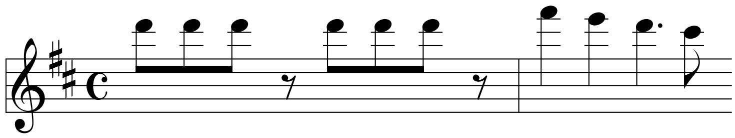


Position	1	2	3	4	5	6	7	8	9	10	11	12
`Pitch` $\otimes$ `Duration`	$\{86,8\}$	$\{86,8\}$	$\{86,8\}$	$\perp$	$\{86,8\}$	$\{86,8\}$	$\{86,8\}$	$\perp$	$\{90,4\}$	$\{88,4\}$	$\{86,4.\}$	$\{85,8\}$
`Interval` $\otimes$ `DurationRatio`	$\perp$	$\{0,1\}$	$\{0,1\}$	$\perp$	$\{0,1\}$	$\{0,1\}$	$\{0,1\}$	$\perp$	$\{4,2\}$	$\{-2,1\}$	$\{-2,1.5\}$	$\{-1,0.\dot{3}\}$

Figure 5.10: Examples of linked viewpoints `Pitch` $\otimes$ `Duration` and `Interval` $\otimes$ `DurationRatio`

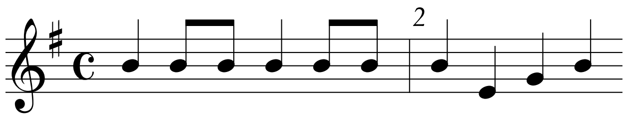
5.2.3.6 Relational Viewpoints

Relational viewpoints, introduced by T. Hedges (2017, p. 224), are viewpoints that take a parameter (or a *referent*, in T. Hedges' terminology). Five relational viewpoints can be seen in Table 5.11, and are described in more detail below. The piece used as the target can be seen in Example 5.2.



Position	1	2	3	4	5	6	7	8	9	10	11	12
`IntervalFirstInTarget`	-15	-15	-15	$\perp$	-15	-15	-15	$\perp$	-19	-17	-15	-14
`IntervalAveragePitchFromTarget`	-14	-14	-14	$\perp$	-14	-14	-14	$\perp$	-18	-16	-14	-13

Figure 5.11: Examples of relational viewpoints `IntervalFirstInTarget` and `IntervalAveragePitchFromTarget`



Example 5.2: The first two bars of Strohbach's *The Drunken Sailor* to be used as the target piece

`IntervalFirstInSource`

`IntervalFirstInSource` is a relational viewpoint, and returns the interval between the current note and the first note in the source piece. Similar to the `Interval` viewpoint, the domain $[\tau]$ of this viewpoint is $\mathbb{Z}$, but is most often restricted to $\{-24, \ldots, 24\}$. Furthermore, the type set $\langle\tau\rangle$ for this implementation (and for all further implementations of relational viewpoints that use intervals) is the basic viewpoint `Pitch`. The viewpoint function is described below in Equation 5.13. Here, R refers to the referent being used, which is the first pitch in the source piece.

$$\Psi_{\text{intervalfirstinsource}}(e^j, R) = \begin{cases} \perp & \text{if } \Psi_{\text{pitch}}(e^j) \text{ is a rest} \\ \Psi_{\text{pitch}}(e^j) - R & \text{otherwise.} \end{cases} \tag{5.13}$$

This viewpoint (and all following viewpoints) is used in this dissertation on its own and part of linked viewpoints such as `IntervalFirstInSource` $\otimes$ `DurationRatio`.

`IntervalFirstInTarget`

`IntervalFirstInTarget` is a relational viewpoint that returns the interval between the current note and the first note in the target piece. The domain $[\tau]$ of this viewpoint is $\mathbb{Z}$ (and is restricted in a similar manner to all previous `Interval` viewpoints. The viewpoint function is similar to the one used in the `IntervalFirstInTarget` function, but R refers to the first pitch in the target piece instead.

`IntervalAveragePitchFromSource`

`IntervalAveragePitchFromSource` is a relational viewpoint that returns the interval between the current note and the average pitch in the source piece. The domain $[\tau]$ of this viewpoint is $\mathbb{Z}$ (and is restricted in a similar manner to all previous `Interval` viewpoints. The viewpoint function is similar to the one described for the `IntervalFirstInSource` function, but R refers to the average pitch from the source piece.

`IntervalAveragePitchFromTarget`

`IntervalAveragePitchFromTarget` is a relational viewpoint that returns the interval between the current note and the average pitch in the target piece. The domain $[\tau]$ of this viewpoint is $\mathbb{Z}$ (and is restricted in a similar manner to all previous `Interval` viewpoints. The viewpoint function is similar to the one described for the `IntervalFirstInSource` function, but R refers to the average pitch from the target piece.

`IntervalAveragePitchFromBothPieces`

`IntervalAveragePitchFromBothPieces` is a relational viewpoint that returns the interval between the current note and the average pitch obtained from both source and target pieces. The domain $[\tau]$ of this viewpoint is $\mathbb{Z}$ (and is restricted in a similar manner to all previous `Interval` viewpoints. The viewpoint function is similar to the one described for the `Interval-FirstInSource` function, but R refers to the average pitch from both pieces.

5.2.4 Contextual Use of Viewpoints

There are several ways in which the viewpoint system as originally described in M. Pearce (2005), Whorley (2013), and T. Hedges (2017) is different to the conceptualisation and the implementation described in the rest of this chapter. For example, in previous implementations in the literature, the viewpoint system was used to generate pieces of music from a large corpus, and generation took place offline. Since in this dissertation the focus is on generating music in response to player interaction in video games, the music generation takes place in an online manner, and the viewpoint system is created from a limited corpus: the source piece (and the target piece when using relational viewpoints).

The differences also extend to the types of viewpoints used. For example, `Interval-FirstInBar` and `IntervalFirstInPhrase` are viewpoints that generate the corresponding intervals of each pitch depending on the first pitch in the bar and in the phrase respectively. However, when generating a transition between two pieces, it is impossible to know whether to use the `IntervalFirstInBar` viewpoint from the source piece or from the target piece.

It is also important to note that some new viewpoints have been added to the system implementation to take into account the use of the target piece in the transition. While it is impossible to know when the transition will take place during the source piece, in this implementation, a musical transition connects to the beginning of the target piece, meaning that the first note of the target piece is always known.

5.2.4.1 Viewpoints Used

Conklin and Witten (1995) reported the best performing system as having an entropy[8] value of 1.87 bits/pitch, and containing the viewpoint features below. These features were generated used the SONG/3 machine learning system, and in this case were used solely to predict Pitch.

- `ScaleDegree` $\otimes$ `Interval`
- `Interval` $\otimes$ `IOI`
- `Pitch`
- `ScaleDegree` $\otimes$ `FirstInBar`

M. Pearce (2005, p. 191) uses a set of viewpoint features referred to as System C, consisting of the features shown below. The system has a cross-entropy[9] value of 1.95 bits/symbol, and was originally developed to minimise cross-entropy over the corpus used.

- `Interval` $\otimes$ `Duration`
- `ScaleDegree` $\otimes$ `IntFirstInPiece`
- `Pitch` $\otimes$ `Duration`
- `ScaleDegree` $\otimes$ `FirstInBar`
- `Interval` $\otimes$ `Tactus`
- `ScaleDegree` $\otimes$ `Duration`

[8] *Entropy*, first described by Shannon (1948) in the context of information theory, is a measure of uncertainty within the system being modelled; the greater the entropy value, the less predictable the encoded messages. Entropy is measured in bits, which describes on average how many bits are needed to transmit each character in a message (and therefore, how predictable the message is).

[9] Cross-entropy is defined by M. Pearce (2005, p. 83) as being "a quantity which represents the divergence between the entropy calculated from these estimated probabilities and the source entropy".

- `Interval` $\otimes$ `DurationRatio`
- `InteralFirstInPiece`
- `Interval` $\otimes$ `FirstInPhrase`

M. Pearce (2005, p. 206) states that System C is not "capable of consistently generating chorale melodies which are rated as equally successful stylistically as those in Dataset 2" (the chorale data set). Therefore, he adds six more viewpoints to create System D, shown below. This system has a cross-entropy value of 1.91 bits/symbol. The features for both System C and System D were generated using a viewpoint selection algorithm based on forward stepwise selection, with the aim of minimising the system's cross-entropy value. Similarly to Conklin and Witten (1995), the viewpoint features for both systems were chosen to generate Pitch alone.

- `Interval` $\otimes$ `Duration`
- `ScaleDegree` $\otimes$ `IntFirstInPiece`
- `Pitch` $\otimes$ `Duration`
- `ScaleDegree` $\otimes$ `Mode`
- `ScaleDegree` $\otimes$ `LastInPhrase`
- `ScaleDegree` $\otimes$ `Duration`
- `Interval` $\otimes$ `DurationRatio`
- `Interval` $\otimes$ `InScale`
- `Interval` $\otimes$ `FirstInPhrase`
- `IntervalFirstInPhrase`

The sets of viewpoint features described above have entropy values calculated based on the corpus used; for Conklin and Witten (1995), this corpus was 100 Bach chorale melodies, while for M. Pearce (2005) this corpus was 185 Bach chorale melodies.

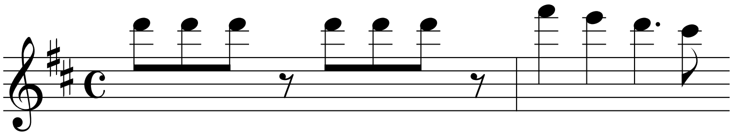
The current context of the viewpoint systems requires the generation of melodies in real-time between two different pre-composed pieces in order to generate a suitable musical transition. Given the difference in context, the use of cross-entropy to determine whether a set of viewpoint features is best suited for the task might not be a good idea. A better approach is to use features that overall describe the suitability of the generated transition (transition success, rate of change, perception, and degree of fitting between pieces). These transition features were used in the first study conducted.

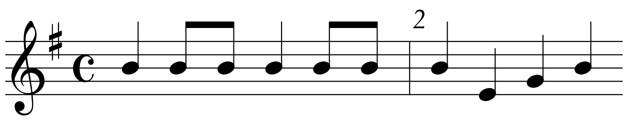
The list below describes the various basic, derived, and relational viewpoints used in this research (and more specifically, in this implementation). A full list of viewpoints used, including linked viewpoints, is given in Table 5.2.

The first two bars of Strauss's *Radetzky-Marsch, zu Ehren des grossen Feldherrn* (Strauss n.d.) is used to illustrate each viewpoint type, and the values for each chosen viewpoint are displayed in a solution array. The referent values used for relational values can be seen below in Table 5.12, with the target piece taken to be the first two bars of Strohbach's *The Drunken Sailor* (Strohbach 1963), as illustrated in Fig. 5.13:

	Source	**Target**
First Pitch	86	71
Average Pitch	86	70
Average Pitch Both Pieces	78	

Figure 5.12: Table of referent values





Position	1	2	3	4	5	6	7	8	9	10
Music Event	4B4	8B4	8B4	4B4	8B4	8B4	4B4	4E4	4G4	4B4

(a) The first two bars of Strohbach's *The Drunken Sailor* to be used as the target piece

Figure 5.13: Referent values used to generate full list of viewpoints

Position	1	2	3	4	5	6	7	8	9	10	11	12
Music Event	8D6	8D6	8D6	8R	8D6	8D6	8D6	8R	4F#6	4E6	4.D6	8C#6
Basic												
Pitch	86	86	86	⊥	86	86	86	⊥	90	88	86	85
Duration	8	8	8	8	8	8	8	8	4	4	4.	8
Octave	6	6	6	⊥	6	6	6	⊥	6	6	6	6
Onset	0	12	24	36	48	60	72	84	96	120	144	180
Derived												
Contour	⊥	0	0	⊥	0	0	0	⊥	1	-1	-1	-1
Interval	⊥	0	0	⊥	0	0	0	⊥	4	-2	-2	-1
DurationRatio	⊥	1	1	1	1	1	1	1	2	1	1.5	$0.\dot{3}$
ScaleDegree	0	0	0	⊥	0	0	0	⊥	4	2	0	11
IOI	⊥	12	12	12	12	12	12	12	12	24	24	36
Relational												
IntervalFirstInSource	0	0	0	⊥	0	0	0	⊥	4	2	0	-1
IntervalFirstInTarget	−15	−15	−15	⊥	−15	−15	−15	⊥	−19	−17	−15	−14
IntervalAveragePitchFromSource	0	0	0	⊥	0	0	0	⊥	4	2	0	-1
IntervalAveragePitchFromTarget	−14	−14	−14	⊥	−14	−14	−14	⊥	−18	−16	−14	−13
IntervalAveragePitchFromBothPieces	−8	−8	−8	⊥	−8	−8	−8	⊥	−12	−10	−8	−7
Linked												
Pitch ⊗ Duration	{86, 8}	{86, 8}	{86, 8}	⊥	{86, 8}	{86, 8}	{86, 8}	⊥	{90, 4}	{88, 4}	{86, 4.}	{85, 8}
Pitch ⊗ IOI	⊥	{86, 12}	{86, 12}	⊥	{86, 12}	{86, 12}	{86, 12}	⊥	{90, 12}	{88, 24}	{86, 24}	{85, 36}
Pitch ⊗ DurationRatio	⊥	{86, 1}	{86, 1}	⊥	{86, 1}	{86, 1}	{86, 1}	⊥	{90, 2}	{88, 1}	{86, 1.5}	$\{85, 0.\dot{3}\}$

Continued on next page

Table 5.2 – *Continued from previous page*

`Octave` ⊗ `Duration`	$\{6,8\}$	$\{6,8\}$	$\{6,8\}$	$\perp$	$\{6,8\}$	$\{6,8\}$	$\{6,8\}$	$\perp$	$\{6,4\}$	$\{6,4\}$	$\{6,4.\}$	$\{6,8\}$
`Octave` ⊗ `IOI`	$\perp$	$\{6,12\}$	$\{6,12\}$	$\perp$	$\{6,12\}$	$\{6,12\}$	$\{6,12\}$	$\perp$	$\{6,12\}$	$\{6,24\}$	$\{6,24\}$	$\{6,36\}$
`Octave` ⊗ `DurationRatio`	$\perp$	$\{6,1\}$	$\{6,1\}$	$\perp$	$\{6,1\}$	$\{6,1\}$	$\{6,1\}$	$\perp$	$\{6,2\}$	$\{6,1\}$	$\{6,1.5\}$	$\{6,0.\dot{3}\}$
`Interval` ⊗ `Duration`	$\perp$	$\{0,8\}$	$\{0,8\}$	$\perp$	$\{0,8\}$	$\{0,8\}$	$\{0,8\}$	$\perp$	$\{4,4\}$	$\{-2,4\}$	$\{-2,4.\}$	$\{-1,8\}$
`Interval` ⊗ `IOI`	$\perp$	$\{0,12\}$	$\{0,12\}$	$\perp$	$\{0,12\}$	$\{0,12\}$	$\{0,12\}$	$\perp$	$\{4,12\}$	$\{-2,24\}$	$\{-2,24\}$	$\{-1,36\}$
`Interval` ⊗ `DurationRatio`	$\perp$	$\{0,1\}$	$\{0,1\}$	$\perp$	$\{0,1\}$	$\{0,1\}$	$\{0,1\}$	$\perp$	$\{4,2\}$	$\{-2,1\}$	$\{-2,1.5\}$	$\{-1,0.\dot{3}\}$
`Contour` ⊗ `Duration`	$\perp$	$\{0,8\}$	$\{0,8\}$	$\perp$	$\{0,8\}$	$\{0,8\}$	$\{0,8\}$	$\perp$	$\{1,4\}$	$\{-1,4\}$	$\{-1,4.\}$	$\{-1,8\}$
`Contour` ⊗ `IOI`	$\perp$	$\{0,12\}$	$\{0,12\}$	$\perp$	$\{0,12\}$	$\{0,12\}$	$\{0,12\}$	$\perp$	$\{1,12\}$	$\{-1,24\}$	$\{-1,24\}$	$\{-1,36\}$
`Contour` ⊗ `DurationRatio`	$\perp$	$\{0,1\}$	$\{0,1\}$	$\perp$	$\{0,1\}$	$\{0,1\}$	$\{0,1\}$	$\perp$	$\{1,2\}$	$\{-1,1\}$	$\{-1,1.5\}$	$\{-1,0.\dot{3}\}$
`ScaleDegree` ⊗ `Duration`	$\{0,8\}$	$\{0,8\}$	$\{0,8\}$	$\perp$	$\{0,8\}$	$\{0,8\}$	$\{0,8\}$	$\perp$	$\{4,4\}$	$\{2,4\}$	$\{0,4.\}$	$\{11,8\}$
`ScaleDegree` ⊗ `IOI`	$\perp$	$\{0,12\}$	$\{0,12\}$	$\perp$	$\{0,12\}$	$\{0,12\}$	$\{0,12\}$	$\perp$	$\{4,12\}$	$\{2,24\}$	$\{0,24\}$	$\{11,36\}$
`ScaleDegree` ⊗ `DurationRatio`	$\perp$	$\{0,1\}$	$\{0,1\}$	$\perp$	$\{0,1\}$	$\{0,1\}$	$\{0,1\}$	$\perp$	$\{4,2\}$	$\{2,1\}$	$\{0,1.5\}$	$\{11,0.\dot{3}\}$
`ScaleDegree` ⊗ `Interval`	$\perp$	$\{0,0\}$	$\{0,0\}$	$\perp$	$\{0,0\}$	$\{0,0\}$	$\{0,0\}$	$\perp$	$\{4,4\}$	$\{2,-2\}$	$\{0,-2\}$	$\{11,1\}$
`ScaleDegree` ⊗ `IntervalFirstInSource`	$\perp$	$\{0,0\}$	$\{0,0\}$	$\perp$	$\{0,0\}$	$\{0,0\}$	$\{0,0\}$	$\perp$	$\{4,4\}$	$\{2,2\}$	$\{0,0\}$	$\{11,-1\}$
`ScaleDegree` ⊗ `IntervalFirstInTarget`	$\perp$	$\{0,-15\}$	$\{0,-15\}$	$\perp$	$\{0,-15\}$	$\{0,-15\}$	$\{0,-15\}$	$\perp$	$\{4,-19\}$	$\{2,-17\}$	$\{0,-15\}$	$\{11,-14\}$
`IntervalFirstInSource` ⊗ `Duration`	$\{0,8\}$	$\{0,8\}$	$\{0,8\}$	$\perp$	$\{0,8\}$	$\{0,8\}$	$\{0,8\}$	$\perp$	$\{4,4\}$	$\{2,4\}$	$\{0,4.\}$	$\{-1,8\}$
`IntervalFirstInSource` ⊗ `IOI`	$\perp$	$\{0,12\}$	$\{0,12\}$	$\perp$	$\{0,12\}$	$\{0,12\}$	$\{0,12\}$	$\perp$	$\{4,12\}$	$\{2,24\}$	$\{0,24\}$	$\{-1,36\}$
`IntervalFirstInSource` ⊗ `DurationRatio`	$\perp$	$\{0,1\}$	$\{0,1\}$	$\perp$	$\{0,1\}$	$\{0,1\}$	$\{0,1\}$	$\perp$	$\{4,2\}$	$\{2,1\}$	$\{0,1.5\}$	$\{-1,0.\dot{3}\}$
`IntervalFirstInTarget` ⊗ `Duration`	$\{-15,8\}$	$\{-15,8\}$	$\{-15,8\}$	$\perp$	$\{-15,8\}$	$\{-15,8\}$	$\{-15,8\}$	$\perp$	$\{-19,4\}$	$\{-17,4\}$	$\{-15,4.\}$	$\{-14,8\}$
`IntervalFirstInTarget` ⊗ `IOI`	$\perp$	$\{-15,12\}$	$\{-15,12\}$	$\perp$	$\{-15,12\}$	$\{-15,12\}$	$\{-15,12\}$	$\perp$	$\{-19,12\}$	$\{-17,24\}$	$\{-15,24\}$	$\{-14,36\}$
`IntervalFirstInTarget` ⊗ `DurationRatio`	$\perp$	$\{-15,1\}$	$\{-15,1\}$	$\perp$	$\{-15,1\}$	$\{-15,1\}$	$\{-15,1\}$	$\perp$	$\{-19,2\}$	$\{-17,1\}$	$\{-15,1.5\}$	$\{-14,0.\dot{3}\}$
`IntervalAveragePitchFromSource` ⊗ `Duration`	$\{0,8\}$	$\{0,8\}$	$\{0,8\}$	$\perp$	$\{0,8\}$	$\{0,8\}$	$\{0,8\}$	$\perp$	$\{4,4\}$	$\{2,4\}$	$\{0,4.\}$	$\{-1,8\}$
`IntervalAveragePitchFromSource` ⊗ `IOI`	$\perp$	$\{0,12\}$	$\{0,12\}$	$\perp$	$\{0,12\}$	$\{0,12\}$	$\{0,12\}$	$\perp$	$\{4,12\}$	$\{2,24\}$	$\{0,24\}$	$\{-1,36\}$
`IntervalAveragePitchFromSource` ⊗ `DurationRatio`	$\perp$	$\{0,1\}$	$\{0,1\}$	$\perp$	$\{0,1\}$	$\{0,1\}$	$\{0,1\}$	$\perp$	$\{4,2\}$	$\{2,1\}$	$\{0,1.5\}$	$\{-1,0.\dot{3}\}$

Continued on next page

Table 5.2 – *Continued from previous page*

`IntervalAveragePitchFromTarget` ⊗ `Duration`	$\{-14, 8\}$	$\{-14, 8\}$	$\{-14, 8\}$	$\perp$	$\{-14, 8\}$	$\{-14, 8\}$	$\{-14, 8\}$	$\perp$	$\{-18, 4\}$	$\{-16, 4\}$	$\{-14, 4.\}$	$\{-13, 8\}$
`IntervalAveragePitchFromTarget` ⊗ `IOI`	$\perp$	$\{-14, 12\}$	$\{-14, 12\}$	$\perp$	$\{-14, 12\}$	$\{-14, 12\}$	$\{-14, 12\}$	$\perp$	$\{-18, 12\}$	$\{-16, 24\}$	$\{-14, 24\}$	$\{-13, 36\}$
`IntervalAveragePitchFromTarget` ⊗ `DurationRatio`	$\perp$	$\{-14, 1\}$	$\{-14, 1\}$	$\perp$	$\{-14, 1\}$	$\{-14, 1\}$	$\{-14, 1\}$	$\perp$	$\{-18, 2\}$	$\{-16, 1\}$	$\{-14, 1.5\}$	$\{-13, 0.\dot{3}\}$
`IntervalAveragePitchFromBothPieces` ⊗ `Duration`	$\{-8, 8\}$	$\{-8, 8\}$	$\{-8, 8\}$	$\perp$	$\{-8, 8\}$	$\{-8, 8\}$	$\{-8, 8\}$	$\perp$	$\{-12, 4\}$	$\{-10, 4\}$	$\{-8, 4.\}$	$\{-7, 8\}$
`IntervalAveragePitchFromBothPieces` ⊗ `IOI`	$\perp$	$\{-8, 12\}$	$\{-8, 12\}$	$\perp$	$\{-8, 12\}$	$\{-8, 12\}$	$\{-8, 12\}$	$\perp$	$\{-12, 12\}$	$\{-10, 24\}$	$\{-8, 24\}$	$\{-7, 36\}$
`IntervalAveragePitchFromBothPieces` ⊗ `DurationRatio`	$\perp$	$\{-8, 1\}$	$\{-8, 1\}$	$\perp$	$\{-8, 1\}$	$\{-8, 1\}$	$\{-8, 1\}$	$\perp$	$\{-12, 2\}$	$\{-10, 1\}$	$\{-8, 1.5\}$	$\{-7, 0.\dot{3}\}$

Table 5.2: List of viewpoints used in this research

5.2.4.2 Transition Region Discretisation

Since the width of the transition region will affect the generated transition as discussed in Section 5.1 (in effect, a wider transition region will result in a longer transition), and the transition will be generated from a combination of the models from both source and target pieces, an additional combination technique must be used together with the viewpoint combination technique. This allows the transition region to slowly ease into music generated by the model for the target piece.

The transition region is first discretised into several smaller sections, and each section is then assigned a ratio that signifies the weights for each corresponding piece. As can be seen in Fig. 5.14, the transition region is discretised into 4 sections, with each one assigned a ratio that shows the probability of a particular piece being chosen.

As an example, the first section is assigned a ratio of $4:1$. This means that in order to generate the next musical event in the transition, the transition system has an 80% chance

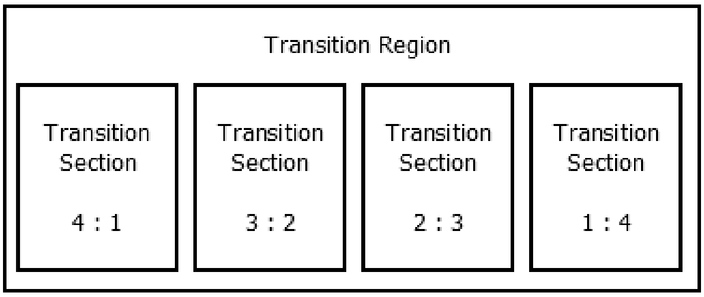


Figure 5.14: A transition region made up of 4 sections, each with their own relevant weighted ratios

of selecting the viewpoint system generated from the source piece, and a 20% chance of selecting the viewpoint system generated from the target piece. As the player traverses further into the transition region, these ratios may change depending on the section they are in.

Therefore, on leaving zone A, the player first enters a section with a ratio of $4 : 1$, meaning that the generated music will be predominantly from the model of the source piece, with a 1 in 5 chance that the generated music would be from the model of the target piece. As the player moves away from zone A and towards zone B, each section contains ratios that represent a decrease in probability that the model from the source piece would be chosen, and an increase in probability that the model from the target piece would be chosen. In the example shown in Fig. 5.14, these ratios are $4 : 1$, $3 : 2$, $2 : 3$, and $1 : 4$ respectively.[10]

This means that in order to determine which model should be chosen in order to generate the corresponding music, one can simply use the ratio that corresponds to the section in the transition region that the player is currently in.

[10]In general, for n sections, the ratios for each section are as follows: $n : 1$, $n - 1 : 2$, $n - 2 : 3$, $\ldots$, $1 : n$.

5.3 Generating Transitions

This section discusses three different techniques that have been used here to generate musical transitions by combining multiple viewpoints. While the first two techniques were found to be unsuitable for generating suitable musical transitions, the third technique was ultimately the one chosen to be used in the study described in Chapter 6. The three different techniques that are described are as follows:

- *weighted averaging*: as described by Wooller and A. R. Brown (2005), this technique generates an event from a sequence of weighted Markov chains based on the player's position within the transition region
- *model selection*: where viewpoint systems for source and target pieces are created, and the next musical event in the transition is generated by selecting a viewpoint system based on the player's position within the transition region
- *weighted viewpoint selection*: where the distributions from viewpoint systems for source and target pieces are combined and weighted, and the next musical event in the transition region is generated based on the resulting distribution

5.3.1 Weighted Averaging

One of the techniques discussed by Wooller and A. R. Brown (2005) in Section 2.3.2.1 is weighted averaging. Here, separate Markov models are created from the source and target pieces. No viewpoints were used when creating the models; instead, each musical event was represented using the same musical representation as discussed in Section 3.3.3.

As previously discussed in Section 5.2.4.2, intermediate Markov chains are created between the source model and the target model in order to generate a suitable transition between the two. Fig. 5.15 shows 5 intermediate steps between the source and target model, with each step composed of its own transition matrix that has weighted transition probabilities depending on its position in the sequence.

	♩D4	♩E4	♩F4	♩G4
♩D4	0.25	0.5	0.25	0
♩E4	0.3	0.3	0.3	0
♩F4	0.5	0	0	0.5
♩G4	0	0	0	0

Model A – 6 : 0

	♩D4	♩E4	♩F4	♩G4
♩D4	0.227	0.5	0.227	0.045
♩E4	0.294	0.294	0.412	0
♩F4	$0.41\dot{6}$	0	0	$0.58\dot{3}$
♩G4	$0.\dot{3}$	$0.\dot{3}$	0	$0.\dot{3}$

5 : 1

	♩D4	♩E4	♩F4	♩G4
♩D4	0.2	0.5	0.2	0.1
♩E4	0.25	0.25	0.5	0
♩F4	$0.\dot{3}$	0	0	$0.\dot{6}$
♩G4	$0.\dot{3}$	$0.\dot{3}$	0	$0.\dot{3}$

4 : 2

	♩D4	♩E4	♩F4	♩G4
♩D4	$0.1\dot{6}$	0.5	$0.1\dot{6}$	$0.1\dot{6}$
♩E4	0.2	0.2	0.6	0
♩F4	0.25	0	0	0.75
♩G4	$0.\dot{3}$	$0.\dot{3}$	0	$0.\dot{3}$

3 : 3

	♩D4	♩E4	♩F4	♩G4
♩D4	0.125	0.5	0.125	0.25
♩E4	0.143	0.143	0.714	0
♩F4	$0.1\dot{6}$	0	0	$0.8\dot{3}$
♩G4	$0.\dot{3}$	$0.\dot{3}$	0	$0.\dot{3}$

2 : 4

	♩D4	♩E4	♩F4	♩G4
♩D4	0.071	0.5	0.071	0.357
♩E4	0.077	0.077	0.85	0
♩F4	$0.08\dot{3}$	0	0	$0.91\dot{6}$
♩G4	$0.\dot{3}$	$0.\dot{3}$	0	$0.\dot{3}$

1 : 5

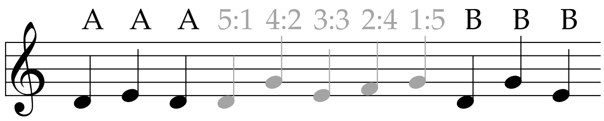
	♩D4	♩E4	♩F4	♩G4
♩D4	0	0.5	0	0.5
♩E4	0	0	1	0
♩F4	0	0	0	1
♩G4	$0.\dot{3}$	$0.\dot{3}$	0	$0.\dot{3}$

Model B – 0 : 6

Figure 5.15: Blended Markov chains that show a weighted average of the transition probabilities between the first model (marked as Model A), and the second model (marked as Model B)

Therefore, a transition matrix weighted in the ratio $4 : 2$ means that when the transition probabilities are calculated, a weighting of 4 is giving to the source model, and a weighting of 2 is given to the target model. Example 5.3 details an example transition of

5 notes generated between two pieces, where each note is generated from the appropriate transition matrix.



Example 5.3: An example of a generated piece using the weighted average transition algorithm. The notes taken from the blended Markov chains are marked in grey. Each note is marked to show which particular Markov chain was used to generate that particular note.

Since a musical transition must be generated in real-time over a transition region, as discussed in Section 5.1, the boundary must be split into smaller subsections with each subsection being associated with a transition matrix.

However, when this approach was implemented, it was found that the generated transition tended to stick too closely to the source piece, and any changes resulted in sudden lurches, making this technique unsuitable for transitions in a real-time environment.

5.3.2 Model Selection

In this technique, two different multiple viewpoint systems are created: the source viewpoint system, which is created from the source piece, and the target viewpoint system which is similarly created from the target piece. The following hand-selected viewpoints were used for each viewpoint system, with the chosen viewpoints based off of the viewpoints used in M. Pearce's (2005) System D. Each order shown below represents the order used for each respective Markov model:

- `IntervalFirstInTarget` (orders 1 to 4)
- `AveragePitchFromTarget` $\otimes$ `Duration` (orders 1 to 4)
- `AveragePitchFromBothPieces` $\otimes$ `Duration` (orders 1 to 4)

- `Interval` $\otimes$ `DurationRatio` (orders 1 to 4)
- `Pitch` $\otimes$ `Duration` (orders 1 to 4)
- `Contour` $\otimes$ `DurationRatio` (orders 1 to 4)

The following viewpoints are used as fallback if the viewpoint system is unable to generate the next relevant attribute. Note that these correspond to the basic types needed to create a musical event, as described in Equation 5.2.

- `Pitch` (orders -1 and 0)[11]
- `Duration` (orders -1 and 0)

The entire transition system is illustrated below in Fig. 5.16

The following steps are taken in order to generate the next musical event:

1. **Select viewpoint system to be used**

   When traversing a transition region, new musical events are generated by selecting a multiple viewpoint system based on the player's position along the transition region. This means that if a player is currently in the transition section closest to the source piece in a transition region containing 4 transition sections, the source viewpoint system is 4 times more likely to be chosen than the target viewpoint system (as can be seen in Fig. 5.14, where the first transition section contains a ratio of $4:1$).

2. **For each basic type in the event space:**

   As stated before in Section 5.2.3.1, the basic types `Pitch` and `Duration` are the basic types used to create the event space used in this dissertation. According to T. Hedges (2017, p. 70), no particular order is given in the

[11] A 0th-order Markov model returns the probability of an event based on its frequency in a sequence, since it is dependent on the previous 0 states that have come before it. A -1th-order Markov model returns an equal probability for each element (T. Hedges 2017, pp. 69–70)

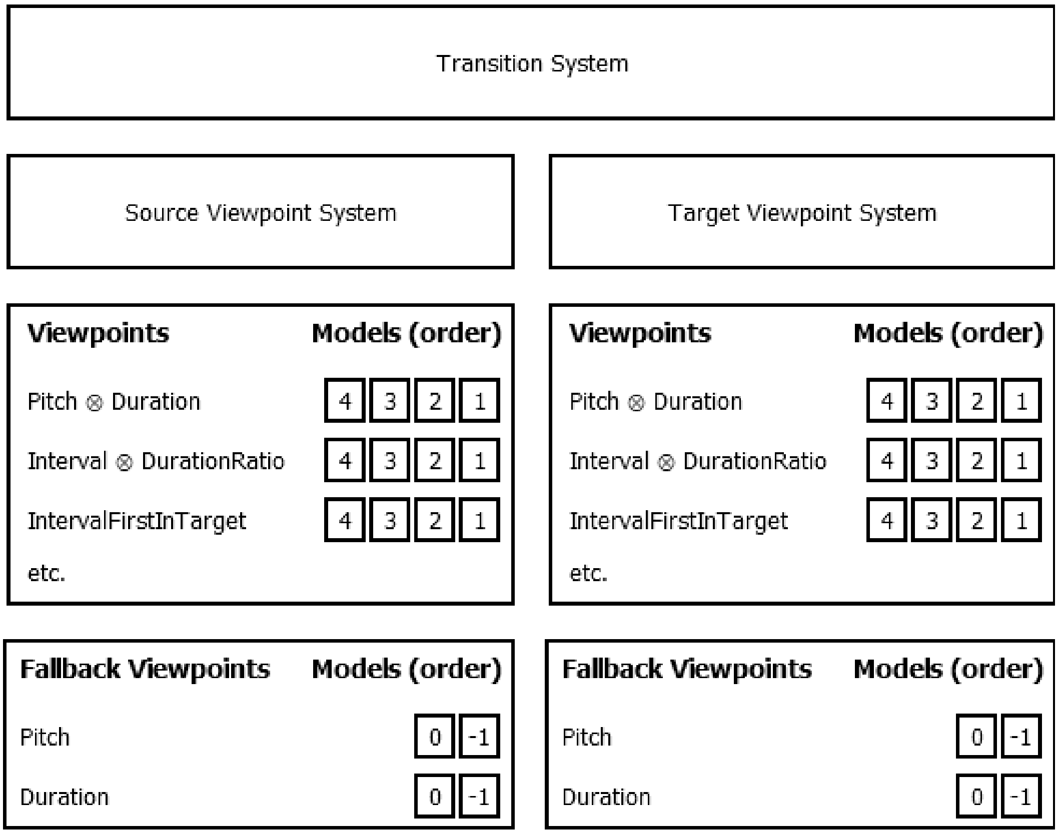


Figure 5.16: Transition system used for the *model selection* transition technique

literature as to which of these basic types should be generated first. In this case, a duration element and a pitch element are generated in that order. This is because if the generated pitch element is `null`, then a rest of that particular duration has been generated and therefore no pitch element is needed. Otherwise, a musical note of the selected pitch and duration has been generated.

(a) **Get global list of elements**

In order to ensure that elements could be generated that appeared in both source and target pieces, as well as elements that could plausibly be generated in a transition between those two pieces, a global list of elements is created.

For example, for the basic type `Pitch`, this is done by generating a list of pitches that start from the lowest pitch from both pieces to the highest pitch from both pieces. This is illustrated below in Fig. 5.17.

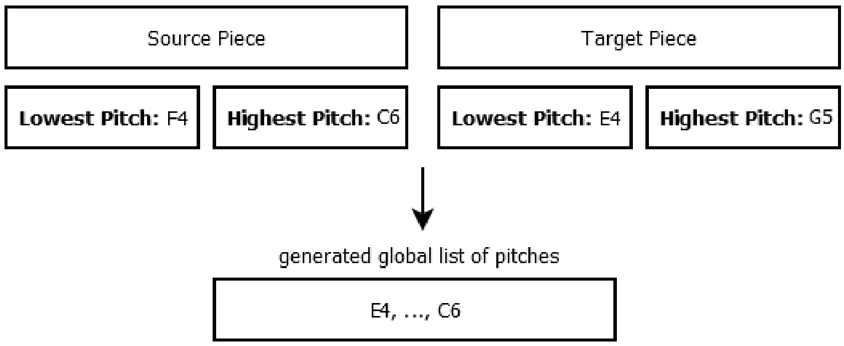


Figure 5.17: Generating a global list of pitches

Similarly, for the basic type `Duration`, this is done by generating a list of durations that start from the shortest duration from both pieces to the longest duration from both pieces, and including dotted durations if they are used.

(b) **For each viewpoint with the current basic type in its type set**

If a particular viewpoint does not have the relevant basic type in its type set $\langle\tau\rangle$, it is unable to predict it and is therefore skipped.

i. **Return probability distribution for current context**

The Markov model associated with the viewpoint is checked to see whether it contains the current context. If the context does not appear in the model, it is not able to predict the next element in the sequence, and is therefore skipped.

ii. **Convert each probable element to a basic type element**

For non-basic viewpoints such as `Interval`, the possible next element in the sequence must be converted into a suitable basic type element in the event space, since they are not immediately usable (as can be seen in Table 5.2). For example, if the previous musical element was 4C4, and the `Interval` viewpoint returned a 50% probability of the next element in the sequence being 2, this needs to be converted into the required pitch representation, which in this example is D4.

(c) **Normalise final probability distribution**

Each element in the final probability distribution is normalised in order to ensure that the sum of probabilities of all elements in the distribution adds up to 1.

(d) **Return element from probability distribution**

Once a final probability distribution is obtained, an element is selected at random (in accordance with the resulting probability distribution) and returned in order to create the final musical element.

In a similar manner to the weighted averaging technique discussed in Section 5.3.1, initial tests showed that this implementation was found to generate material from the source piece the majority of the time, and only generate material from the target piece in a sudden and abrupt manner. This meant that this technique was unsuitable for generating musical transitions in a real-time environment.

5.3.3 Weighted Viewpoint Selection

This technique combines elements from both the weighted averaging technique described in Section 5.3.1, as well as the model selection technique described in Section 5.3.2.

For each basic type in the event space, each of the model orders associated with each viewpoint is assigned a level of certainty, obtained by measuring the model's Shannon entropy for each basic type as shown in Equation 5.14.

$$H(p_m) = -\sum_{t\in[\tau]} p_m(t)\log_2 p_m(t) \tag{5.14}$$

Note that in the case of $p_m(t) = 0$, since the full expression evaluates to $0 \log_2 0$, the result is taken to be 0 since $\lim_{x\to 0} x\log_2 x = 0$.

To generate the next musical event using this technique, the following steps are taken:

1. **For each basic type in the event space:**

   In a similar manner to previous techniques, the basic types `Pitch` and `Duration` are used to create the required event space.

   (a) **Viewpoint selection from source system**

   In a similar manner to previous techniques, the Markov model associated with the viewpoint is checked to see whether it contains the current context. If the context does not appear in the model, it is not able to predict the next element in the sequence, and is therefore skipped. The remaining viewpoints will be used in order to compute a probability distribution as described later.

   (b) **Viewpoints selection from target system**

   The same procedure used for the source system is used here, as described in point 1a above.

   (c) **Generate source and target probability distributions**

   In this step, all the chosen viewpoints obtained from a particular viewpoint system must be combined in order to assign a probability for each possible element to be generated for the relevant basic type. This is performed for both the source and target viewpoint systems, resulting in two different probability distribution.

A probability distribution is generated by using the *weighted geometric combination function*[12] and is described below in Equation 5.15.

$$p(t) = \frac{1}{R}(\prod_{m \in M} p_m(t)^{w_m})^{\frac{1}{\sum_{m \in M} w_m}} \tag{5.15}$$

The model weight is represented by w_m and is defined in terms of relative entropy as follows in Equation 5.16:

$$w_m = H_{\text{relative}}(p_m)^{-b} \tag{5.16}$$

Here, b is a bias factor used to give preference to models with lower entropy (and therefore, models that are more certain) by using an exponential weighting, and is defined by T. Hedges (2017, p. 69) as $b \in \mathbb{Z}^*$. With a bias factor value of 0, no weighting takes place. In this dissertation, $b \in \mathbb{Q}$ as a bias value of 1.4 is used, which was acquired through trial and error by listening to the generated results.

Relative entropy is defined in Equation 5.17 below in terms of maximum entropy and the model's Shannon entropy (defined earlier in Equation 5.14).

$$H_{\text{relative}}(p_m) = \begin{cases} \frac{H(p_m)}{H_{\text{max}}(p_m)} & \text{if } H_{\text{max}}(p_m) > 0 \\ 1 & \text{otherwise.} \end{cases} \tag{5.17}$$

[12]The weighted geometric combination function is introduced by M. Pearce (2005, p. 116) and is presented as an improvement to the previously used weighted arithmetic combination function, which is not discussed here.

Finally, the maximum entropy of a distribution is defined as follows in Equation 5.18:

$$H_{\max}(p_m) = \log_2|[\tau]| \tag{5.18}$$

(d) **Combine to create final probability distribution**

To generate a new musical event using this technique, both probability distributions generated from the source and target viewpoints are combined. This is done by using the appropriate weighting ratio (as shown in Fig. 5.14) that is dependent on the player's location along the transition region. An example of this can be seen in Table 5.3. Here, the first column (marked *Duration*) shows the list of duration elements that can be generated, while the columns marked *Source* and *Target* show the probability distribution of duration elements for the source and target pieces. These probability distributions are then combined (in this case, using a $2:3$ ratio) to form the combined distribution shown in the final column.

Duration	**Source**	**Target**	**Combined**
4	1	0.941	0.965
4.	0	0	0
2	0	0.0582	0.0349
2.	0	0	0
1	0	0	0

Table 5.3: Combined probability distributions from the source and target viewpoint systems using a $2:3$ ratio

(e) **Return element**

Once a combined probability distribution is obtained, an element is selected at random and returned in order to create the final musical element.

This technique is used to generate musical transitions in a study described in Chapter 6.

5.4 Conclusions

This chapter has discussed issues with zone boundaries in video games and introduced the concept of a transition region, allowing the use of generative algorithms for musical transitions. It has also introduced Markov models and their use in a multiple viewpoint system, as well as the specific system used to generate transitions within a video game.

We may remember . . . that many years ago Marcel Duchamps [sic] allowed a number of threads to fall on pieces of cloth and then framed and preserved them. Our example shall be in the field of music.

John R. Pierce (as J. J. Coupling), "Science for Art's Sake", p. 90[1]

6 Generative Transitions in a Game Environment

This chapter describes in detail the system design, methodology, and dataset used for a study investigating the use of the new transition technique in the context of a video game environment. This is compared to the crossfading technique, which is frequently used in the industry to transition between different pieces of music. Section 6.1 describes the aims and objectives of this study, and Section 6.2 details the design of the system used to run the study. Section 6.3 describes the dataset used in the study, as well as the

[1] John R. Pierce (Nov. 1950). "Science for Art's Sake". In: *Astounding Science Fiction*. Ed. by John W. Campbell Jr., pp. 83–92

generation of a lookup table that is able to select an appropriate viewpoint set to use to generate a transition. Finally, Section 6.5 describes the methodology used in the study.

6.1 Aims and Objectives

The primary objective of the study is to investigate the use of a new transition technique in the context of a video game environment. In particular, this study aims to answer the following questions:

- Can multiple viewpoint systems be used to generate musical transitions in a game environment?
  - Are multiple viewpoint systems suitable for use in real-time environments such as games?
  - How successful are these generated transitions?
  - Are some viewpoints better than others at generating musical transitions?
- Can generated transitions increase the amount of immersion players experience in games?

The custom transition technique will be compared with the crossfading transition, for two reasons. First of all, the crossfading technique is frequently used in the industry as a standard transition technique due to its ease of implementation. Secondly, it is used due to its ability to deal with varying lengths of transition times; here ideally, the custom technique will be able to fulfill a similar role in a more musically sophisticated manner.

6.2 System Design

In a similar manner to the first study, music was generated using the *midi-dot-net* library to schedule MIDI messages and the default software synthesiser *Microsoft GS Wavetable SW Synth*, as was previously done in the study described in Chapter 3.

6.2.1 Game Environment Implementation

The game environment used for this study was built in the Unity[2] game engine using C#. All assets created for the game were found online for free or created specifically for the environment; this included 3D models, textures, particle systems, shaders, and sound effects. Proper attribution for these assets can be found in Appendix E.

A low-poly[3] graphical style was used to implement the game environment; this style is noted for its use of minimalism by taking complex objects and constructing them using simpler geometric shapes. There have been many successful games released in this style, such as underwater adventure game *Abzû* (Giant Squid Studios 2016), the FPS/puzzle game *Superhot* (Superhot Team 2016), and the puzzle game *The Witness* (Thekla Inc. 2016). Using this style allows the game environment to be more visually cohesive, especially when using pre-made assets, since it does not need to be built with realistic object models that might differ in quality.

The environment was divided into six distinct islands, with each island being designed specifically to evoke a certain theme or idea. The islands used and other aspects of the game environment are described below. A short brief is included (given in italics), intended to describe the scene to the composer.

[2]Unity is a 3D game engine that allows for the development of video games. It is developed by Unity Technologies and can be found at `http://www.unity.com`. At the time of writing, is released under 3 different licenses: pro, plus, and personal. This dissertation uses the personal license, which provides a free version of Unity.

[3]*Low-poly* is short for "low polygon count".

6.2.1.1 Global Elements

Figure 6.1: The player character, bridges, and the ocean

As can be seen in Fig. 6.1, several elements in the environment such as the ocean, the player character, and bridges, are common to all six islands and therefore an essential part of the game environment.

ocean

The ocean is a third-party package from the Unity asset store, and is implemented as a large flat plane containing several vertices that are spaced evenly along the plane. The corresponding shader gives the appropriate colour to each segment in the ocean, as well as the surf found between the ocean and each island. This shader was chosen to match the low-poly graphics of the entire game environment. Finally, a script manipulates the position of each vertex, moving these up and down to create the illusion of a moving body of water.

player character

The player character is the avatar that represents the player inside the game environment, and is made up of a number of components. The character model is humanoid, and is animated to allow for idling, crouching, walking, running, jumping, and falling; an appropriate animation controller handles the transitions between each of these states. The model also contains a capsule collider[4], used for collision detection. Finally, the character model is a rigid-body, meaning that within the physics simulation in the game, the character model has mass and obeys gravity.

An audio source allows for walking sounds to be played that are synchronised to the character's footsteps in an animation sequence. These sounds are also dependent on the environment the player is in, meaning that there are unique footstep sounds for all six islands as well as the bridges. These sounds are implemented using round-robin sampling, where multiple unique sounds are associated to a particular environment, and the next sound to be played is chosen at random.

The game camera, which is positioned just behind the player character, uses a perspective projection and implements occlusion culling (where any object that is being hidden by other objects in the scene are simply not rendered). Furthermore, a script prevents the camera from clipping[5] into other models in the environment.

A script allows for control of the player character. In a similar manner to other games in a 3D environment, the character can be moved using the `WASD` keys or arrow keys, and the camera's position can be changed by moving the mouse, allowing the player to look around the environment by having the camera rotate around the character model.

[4]In game development, a collider is a shape that is placed around the object that is used during collision detection; whether two objects have physically collided. The size and shape of the collider determines how often collision detection is called and how accurate collision is. In this case, a capsule collider is one shaped like a capsule and is a suitably efficient middle-ground between speed and accuracy.

[5]Camera clipping is a common problem in game development where the game camera is forced inside an object model resulting in visual glitching.

bridges

The bridges allow players to cross over from one island to another, and are made up of four segments. The sides of each bridge are covered with an invisible barrier, so as to prevent players from falling off the bridge into the water below.

The game environment was lit by using a large directional light that acted like the sun. Several other light sources can be found across the islands, such as campfires and torch light. The *Baked Indirect* lighting mode was used, allowing a combination of real-time lighting that allowed for dynamic shadows (such as those cast by the player or other moving objects), as well as the use of baked lighting, where shadows for static objects are pre-calculated.

Each object in the game environment contains an appropriate collider (normally a mesh collider or a capsule collider) used to prevent the player character from walking through objects by detecting collisions. Each island was also surrounded by invisible colliders in order to prevent players from walking off the island.

6.2.1.2 Forest Island

> *"This island is heavily forested, and inhabited by various types of wildlife and plants. This island is uninhabited except for the logger on the far side of the island, but is often frequented by campers."*

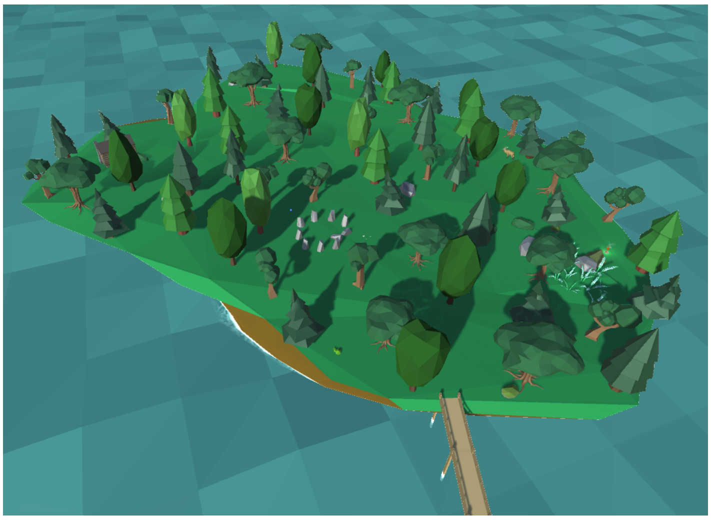


Figure 6.2: Forest Island

Many games contain forested areas or woods as unique locations. Examples of such areas include Channelwood from the puzzle game *Myst* (Cyan 1993), Macalania Woods from the JRPG *Final Fantasy X* (Square 2001), and the Sasau Forest in the action/RPG *Kingdom Come: Deliverance* (Warhorse Studios 2018).

In this implementation, the forest island seen in Fig. 6.2 contains several different varieties of trees, bushes, plants, and other types of flora. The island is also home to a family of foxes and moose. There are three unique locations on the island:

- a wooden cabin found on the far side of the island, home to a lumberjack
- a stone circle found in the middle of the forest, where the occasional fairy may appear
- a tent and a campfire found under a ridge on the eastern side of the island

The fairy, campfire, and lumberjack's torchlight were all implemented using individual particle systems (or a combination of systems). Fire sources, such as the campfire or the torchlight, emitted their own light, and also had audio sources associated with them allowing them to produce sound effects such as the crackling of fire.

6.2.1.3 Village Island

> *"This island hosts a small farming and fishing community that live in a quiet village. The locals enjoy spending their time by the lake or in the small grove of trees on the far side of the island."*

Figure 6.3: Village Island

Similarly, many games take place in either large sprawling urban environments or small medieval villages. Such examples include Pelican Town from the farming simulation game *Stardew Valley* (Barone 2016), Woodtick from the point-and-click adventure game *Monkey Island 2: LeChuck's Revenge* (LucasArts 1991), and Pallet Town from JRPG *Pokémon Red* (Game Freak 1996).

In this environment, the village island seen in Fig. 6.3 contains a small village at the far side of the island. A small farm containing pigs, sheep, and cows, as well as a variety of crops, provides food for the locals. Several small groves of trees can be found towards the eastern side of the island, and a large lake can be found in the centre of the island.

6.2.1.4 Snow Island

> *"A harsh winter has descended on this island, and the ground is covered with inches of snow. The island is uninhabited, save for the lighthouse keeper on the far side of the island, and the local fauna."*

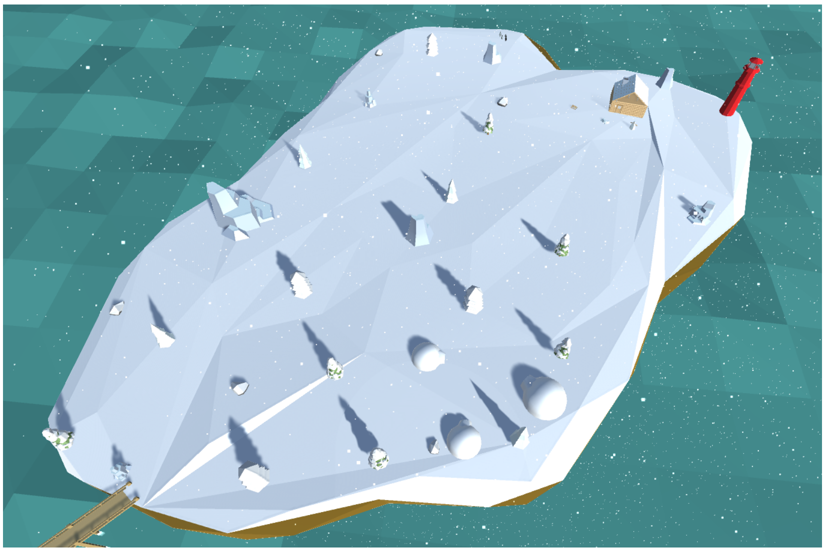


Figure 6.4: Snow Island

There are several examples of icy and arctic environments in games. Such examples include the entirety of the puzzle/platformer *Never Alone* (*Kisima Inŋitchuŋa*) (Upper One Games 2014), the mountain in the adventure game *Journey* (thatgamecompany 2012), and Freezeezy Peak in the 3D action/platformer *Banjo-Kazooie* (Rare 1998).

Here, the snow island seen in Fig. 6.4 is rather sparse, and populated mainly by icy boulders, rocks, and trees that have been snowed over. A lighthouse can be found at the far end of the island, along with a large log cabin. A group of penguins can also be found on the western side of the island. The island also contains falling snow to add to the atmosphere; this was implemented using a particle system.

6.2.1.5 Circus Island

> *"The circus has come to town, and set up a variety of stalls on this island. Visitors can expect a grand spectacle, with acrobats, death-defying spectacles, and a firework show."*

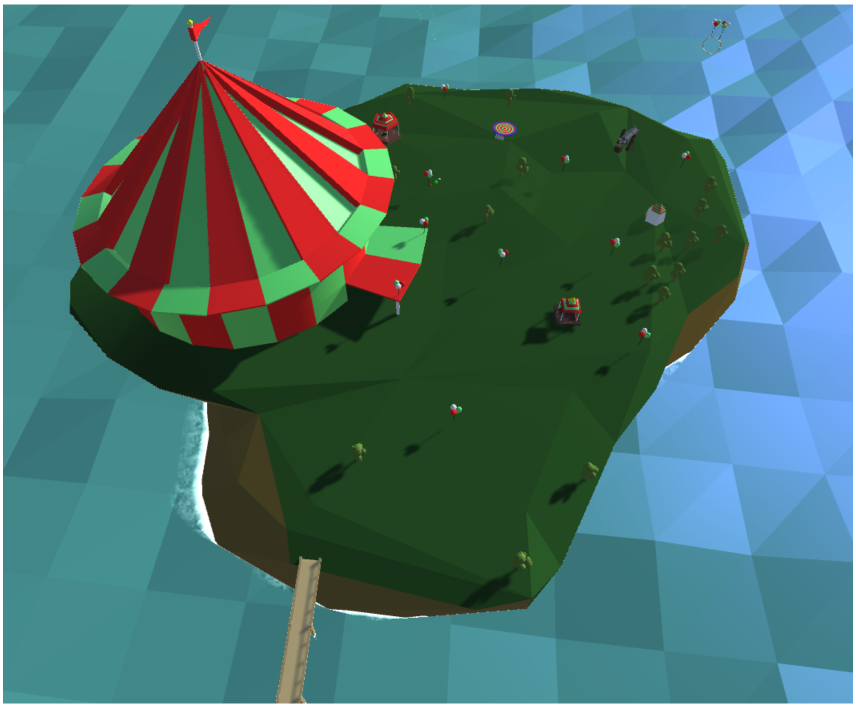


Figure 6.5: Circus Island

While a carnival or circus environment is not particularly common in games, there are still a few examples that may be noted. These include Darkmoon Faire in the MMORPG *World of Warcraft* (Blizzard Entertainment 2004), the Carnival of the Damned in the point-and-click adventure game *The Curse of Monkey Island* (LucasArts 1997), and Flotsam's circus in Palm Brinks in the action game *Dark Chronicle* (Level-5 2002).

On this island as seen in Fig. 6.5, players can find a large carnival tent; walking inside reveals a seating area and a juggling display. Outside, several attractions can be seen dotted around the island; these include a merry-go-round, a trampoline, a acrobat's cannon (with the corresponding hoop target), a knife-throwing stall, and a food stand. Several trees and balloons can be found around the island, with larger hot-air balloons found higher up in the sky. A fireworks display also takes place above the carnival tent.

The fireworks were implemented using three individual particle systems to represent the launching of the firework into the sky, the light display, and trails given off after the firework has gone off. Each firework contained an audio source which was used for sound effects associated with the launching of the firework into the air and the fizzling sound of the firework's light display. The carnival tent also contains a light source that serves as a spotlight onto the performance taking place inside.

6.2.1.6 Desert Island

> *"This island is a harsh desert with little to no wildlife, and sparsely populated with various cacti. It hosts a small community of traders on the far side of the island, and a tomb of a great king can be found in the centre of the island."*

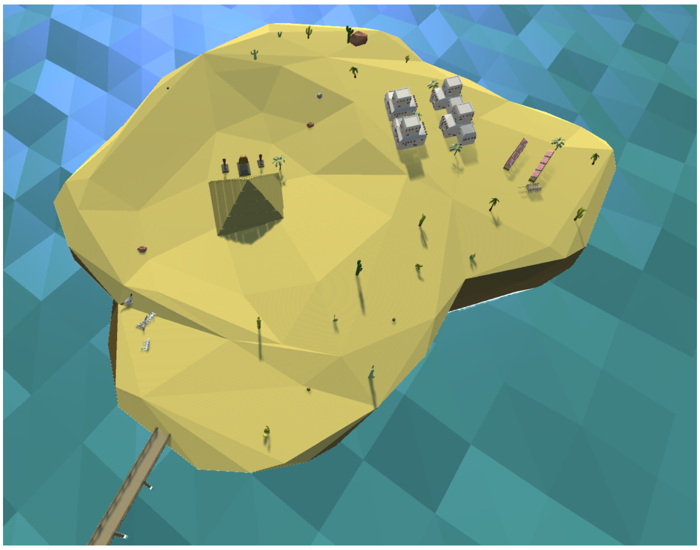


Figure 6.6: Desert Island

Although not many games take place entirely in the desert, there are several games that make use of them. Such examples include the desert in the adventure game *Journey* (thatgamecompany 2012), Bikanel Island in the JRPG *Final Fantasy X* (Square 2001), and the desert ruins in the puzzle game *The Witness* (Thekla Inc. 2016).

In this implementation, the island as seen in Fig. 6.6 is a rather sparse sandy desert, dotted with various different cacti and brown rocks. A large pyramid stands in the centre, with several monuments found to one side. A desert village can be found at the far side of the desert, with a market place close by. A pile of skulls and bones can also be found on the opposite side of the island.

6.2.1.7 Spooky Island

> *"This island has been afflicted by a terrible curse, and has been plunged into a state of constant fog, allowing terrible creatures to live here. Only the bravest of people visit this island, and only the local gravedigger lives here."*

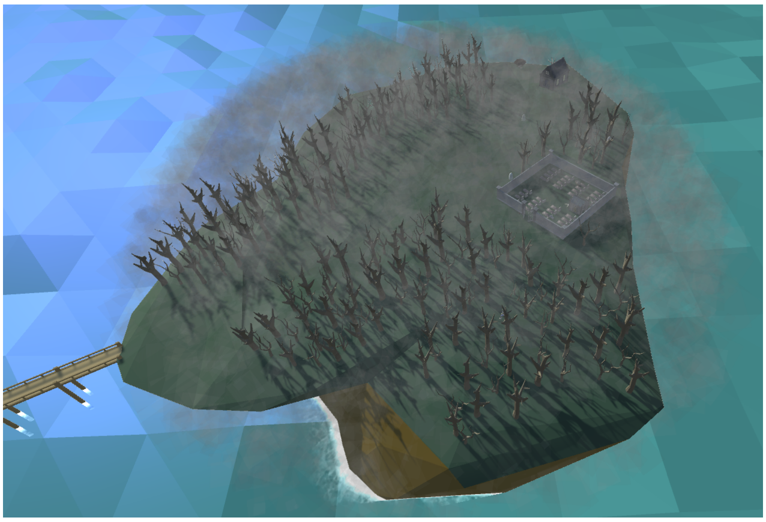


Figure 6.7: Spooky Island

Several games (such as horror games, or games that contain horror elements) are entirely designed to scare players. Other games may contain specific locations that are intended to be creepy to players (such as graveyards or catacombs). Such examples include the region of Karazhan in the MMORPG *World of Warcraft* (Blizzard Entertainment 2004), the Shadow Temple in the action/adventure game *The Legend of Zelda: The Ocarina of Time* (Nintendo EAD 1998), and the Mad Monster Mansion from the 3D action/ platformer *Banjo-Kazooie* (Rare 1998).

Here, the island seen in Fig. 6.7 is covered in a fog and a large section of dead trees separated by a path. The island also contains a cemetery, containing a mausoleum and a number of graves, as well as a dark and foreboding house. The island is also populated with werewolves, zombies, and ghosts, and is also covered with mushrooms.

Similarly to the forest, the campfire is made up of a combination of particle systems, a light source, and an audio source for associated sound effects. The island's fog is implemented using a particle system that is consistent with the game environment's low-poly style. The zombie models are also animated; one can be seen trying to crawl out from under the mausoleum, the head of another can be seen moving about inside a grave, and the last zombie can be seen idling amongst the trees on the east side of the island. All zombies have audio sources associated with them in order for them to emit various sound effects such as moans and growls. Each ghost is also associated with an animator, allowing it to slowly bob up and down.

6.2.2 Integration of the Transition Algorithm

This section discusses the implementation and integration of the transition algorithm discussed in Chapter 5 for use within in game environment.

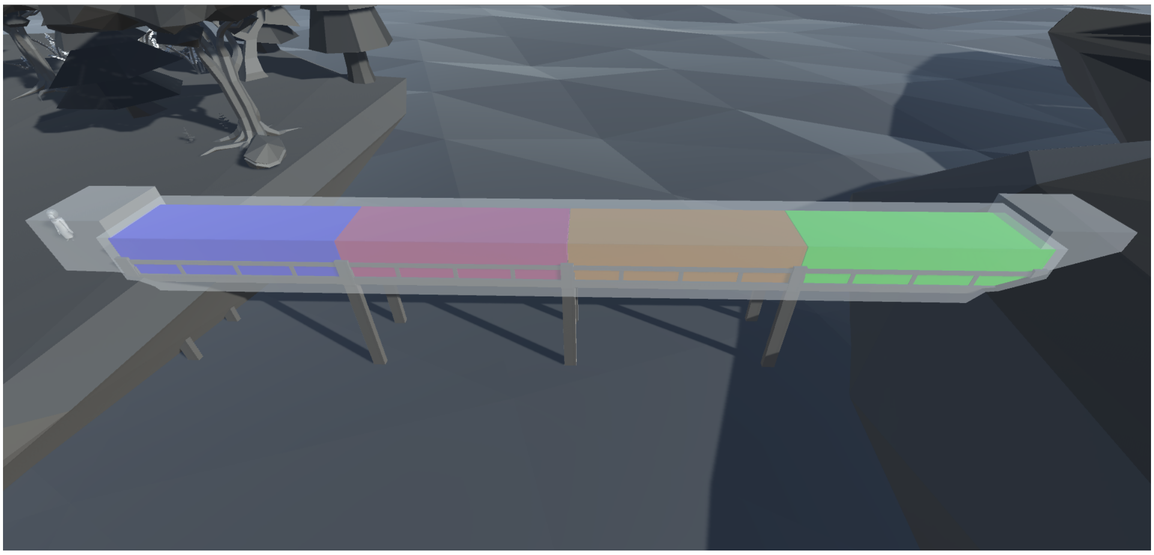


Figure 6.8: Breaking up the transition area into cubes

6.2.2.1 Implementing the Transition Region

A possible approach to implementing the transition region, as discussed in Section 5.1, is to subdivide it into equal parts. In this game environment implementation, the bridges found between islands have been used as the transition regions. An example is shown in Fig. 6.8, showing a bridge between the forest island and the carnival island that is divided into four equal parts. While these parts have been coloured in the illustration and the rest of the game environment rendered in greyscale for convenience, they are not actually visible during run-time. Note the difference between the island cubes (in white and on either end of the bridge), and the transition cubes (given the colours blue, red, orange, green); while transition cubes are generated automatically based on the size of the transition region, the island cubes are placed when building the environment and used to connect two islands together by placing a bridge in between.

By dividing the transition region into a number of parts, the system can easily weight the transition algorithm accordingly depending on the player's position within a region.

However, to keep track of where the player is at any given time during the transition, an event system has been implemented; this can be seen in Fig. 6.9.

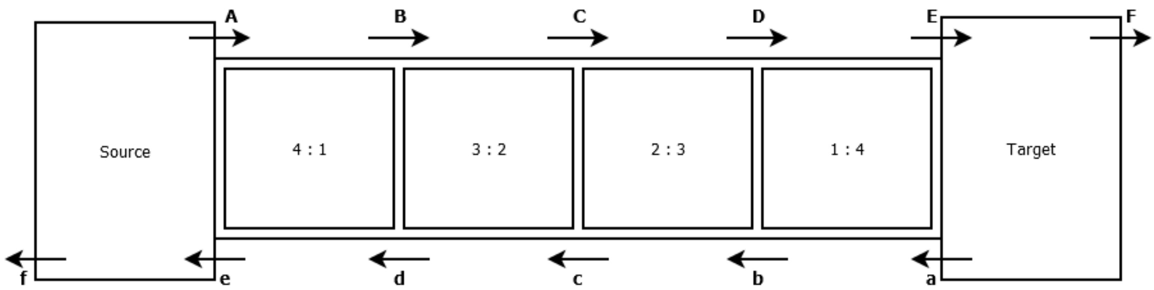


Figure 6.9: Transition triggers and their corresponding events

Each individual cube shown in Fig. 6.9 (both island cubes and transition cubes) is implemented in Unity as a trigger zone; this allows interactions between the cube and the player character to be handled via collision detection. Two such interactions are `OnTriggerEnter` and `OnTriggerExit`, which are described further below.

`OnTriggerEnter`

This event marks the moment that a collision take a place, no matter how slight.

If this event is triggered when entering an island cube, nothing happens. However, if this event is triggered when entering a transition cube, the system immediately discards any material it had previously generated and scheduled, and proceeds to generate and schedule new material based on which transition cube the player has now entered.

`OnTriggerExit`

This marks the moment that a collision is no longer taking place.

If this event is triggered when exiting an island cube, a check is made to determine if the player is stepping onto the bridge (marked **A** or **a** in Fig. 6.9), in which case the system should begin playing generated transition music from the first transition cube, or leaving the bridge (marked **E** or **e** in Fig. 6.9), in which case the piece corresponding to the island is scheduled to be played.

However, if the event is triggered when exiting a transition cube while remaining on the bridge (marked **B**, **C**, **D**, **b**, **c**, and **d** in Fig. 6.9), the system simply updates the player's current position to be that of the current cube. This is necessary since this particular case also triggers an `OnTriggerEnter` event, which needs to know which cube the player is in.

In Section 5.1, several player behaviours were introduced as being important to consider when implementing transition regions. Each player behaviour is discussed below with reference to their implementation in this study.

seamless transition between zones A and B (Fig. 5.3a)

This figure illustrates behaviour that involves the player entering the transition region from the source island and exiting the region into the target island, regardless of the path that is taken in between the two islands. This is considered to be standard player behaviour that allows for a musical transition between two zones.

entering the transition region from zone A, and returning to zone A (Fig. 5.3b)

This figure illustrates behaviour that involves the player entering the transition region from one zone, and leaving the transition region to the same zone (in effect, not crossing the transition region). While this is technically implemented within the game environment, this player behaviour is restricted for the purposes of the study in order for players to experience a complete musical transition from zones A to B; this is done by hiding the source island and blocking off the player's path.

entering the transition region from zone A, and remaining inside it (Fig. 5.3c)

This figure illustrates behaviour that involves the player traversing parallel to the transition boundary instead of crossing it. While this is implemented within the game environment, the player is restricted by the width of the transition region (which in this case, corresponds to the width of the bridge).

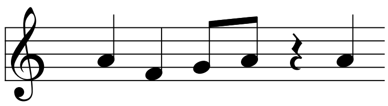
entering the transition region from zone A, and stopping inside it (Fig. 5.3d) This figure illustrates behaviour that involves the player entering the transition region and stopping in the middle of it. This behaviour may be common in open-world games where the transition regions aren't defined as clearly. This behaviour is catered for within the game environment as well as the study, but participants will not be able to continue with the study until they have crossed the transition region into the target island.

6.2.2.2 Generating Notes and Preparing them to be Played

In order to prevent the transition algorithm from being called after each generated note, the algorithm generates a sequence of musical events that are then batched. The current implementation generates 20 musical events in one batch. Each generated note is converted to a corresponding `NoteOnMessage` and `NoteOffMessage` MIDI messages, while each generated rest is similarly converted to a corresponding `RestOnMessage` and `RestOffMessage` MIDI messages.[6] Both `NoteOnMessage` and `RestOnMessage` are called to indicate that the respective musical event will be begin playing at a particular time, while the `NoteOffMessage` and `RestOffMessage` message indicate the end of their respective musical event. Finally, at the end of the generated sequence of musical events, a `CallbackMessage` is added that allows the transition algorithm to be called

[6]While the `RestOnMessage` and `RestOffMessages` do not appear in the MIDI specifications, they were created by extending the base `ChannelMessage` in order to make things easier to work with.

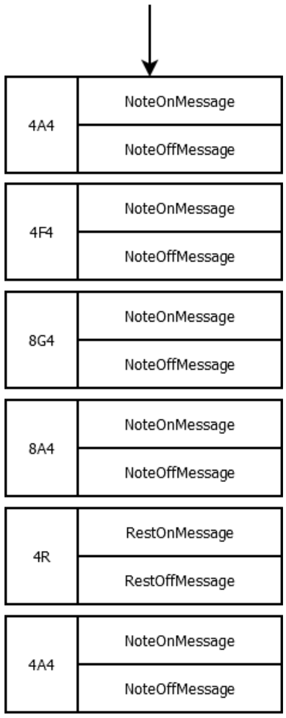
again to generate the next batched sequence of musical events. This process is displayed below in Fig. 6.10.



(a) Musical representation of generated events

```
4A4, 4F4, 8G4, 8A4, 4R, 4A4
```

(b) Internal representation of generated events



(c) Sequence of generated MIDI messages (with corresponding notes or rests for reference)

Figure 6.10: Batching note generation

A frequent occurrence is a transition taking place before the respective `NoteOffMessage` has been sent. This means that the scheduled pitch triggered with the previous `NoteOnMessage` will not end due to the lack of the corresponding `NoteOffMessage`. This results in notes that continue to ring while new notes are being scheduled and played. This issue is illustrated below in Fig. 6.11. To solve this, the corresponding `NoteOffMessage` is generated using the appropriate clock time before any of the transition notes are generated.

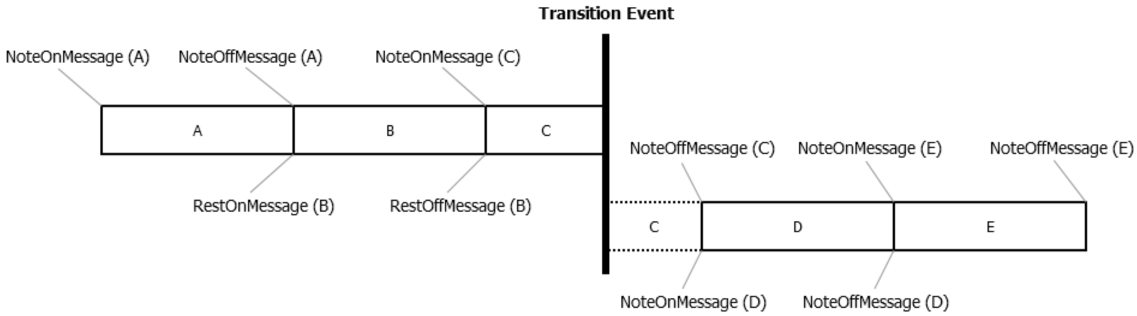


Figure 6.11: Issues with sustained notes

6.2.3 Study Session Generation

For each pair of islands presented to participants, the following steps were followed in order to make sure that both islands are suitably connected and that a walkable path is created between the source and target islands over a bridge.

1. In order to determine where the target island should be placed, the source island and its associated bridge are first spawned into the environment and their positions are recorded.

2. A vector is taken to determine the direction the bridge is facing, and the target island is placed at the end of the bridge facing the source island (using the direction vector as a reference).

3. Finally, the target island is rotated using quaternion rotation in order to ensure that both islands match up accordingly.

6.3 Dataset

The music pieces created for the study were commissioned and composed by Jan Flessel[7], and have been included in Appendix E.

[7] Jan Flessel is a German composer who, at the time of writing, is based in Copenhagen, Denmark. His website can be found at `https://www.janflessel.com/`.

Before sitting down to write anything, the composer was presented with visuals of all 6 islands that were to be used in the study (as described in Section 6.2.1), and was told that these would work in the context of a game environment. The list below shows certain keywords that refer to moods and areas that the composer thought were appropriate to convey in the pieces he created for each island.

- **Snow**: Cold, but warm in atmosphere due to the snowman and the sled. No sign of death. Shelter.
- **Carnival**: Fun. Children. Celebration. Silliness
- **Desert**: Signs of death. Ruins and wonder. Yet contains life due to the market and town.
- **Spooky**: Desolation. Abandoned. Misty, yet the graveyard provides some silliness and fun elements. Chapel could be shelter or place of wonder as well.
- **Forest**: Elvish mood. Some Wisps. Ruins, but barely recognisable. Peaceful. Home.
- **Village**: Grass. Silence. Distance.

After being presented with the written description for each island (shown in Section 6.2.1), several ideas were discussed on how to best compose such pieces, such as by leaning into (or avoiding) particular musical tropes associated with different environments[8]. Such ideas included the use of high pitched notes for the Snow island, the use of longer notes for the Spooky island, and avoiding cliches in the Desert island.

The composer was asked to write each piece in common time using a tempo of 120 beats per minute and in the scale of C major, though the use of different modes was allowed. Changes in time signature, key signature, tempo, and the use of dynamic

[8]An introduction to the use of musical tropes in video games can be seen in "Soaring Through the Sky: Topics and Tropes in Video Game Music" (Atkinson 2019).

markings was not allowed. In a similar manner to the study described in Chapter 3, performance markings such as ties were ignored; this is because the latter would interfere with the generation of musical events during the actual transition.

Each piece was to be composed as a simple melody that could be used in a video game (meaning that it should be able to loop if necessary), and was to be around 2 minutes long. The composer was told that these pieces would ultimately be played using the default software synthesiser *Microsoft GS Wavetable SW Synth* (as discussed in 6.2), and was allowed to utilise the full range of the piano. Twelve pieces were written in total, resulting in two pieces for each island.

In a similar manner to the first study (and as described in Section 3.3.3), a 240 piece dataset was created by taking all possible combinations of six source island types, five target island types[9], two transition types, and the two types of source and target pieces. This dataset was created offline before the study took place and divided into 16 batches. A study participant would be presented a specific batch in a randomised order.

6.4 Creating a Lookup Table

For this study, a lookup table was used containing different sets of viewpoints. This allowed different viewpoint sets to be chosen to generate suitable transitions depending on the source and target pieces in the game environment.

To create such a lookup table, the twelve composed pieces were first grouped into separate categories based on the length of notes used (referred to as the **Tempo** group, and grouped into the three categories **Fast**, **Medium**, and **Slow**) and based on the pitches used (referred to as the **Octave** group, and grouped into the three categories **High**, **Medium**, and **Low**). This categorisation can be seen in Table 6.1 below.

[9] A transition between the same source and target island is not taken into account here, since this does not make sense within the context of a game environment.

Piece	Tempo Category	Octave Category
Forest 1	Fast	High
Forest 2	Medium	High
Village 1	Medium	High
Village 2	Medium	High
Desert 1	Medium	Medium
Desert 2	Medium	Medium
Spooky 1	Slow	Low
Spooky 2	Slow	Low
Snow 1	Slow	High
Snow 2	Medium	High
Circus 1	Fast	High
Circus 2	Fast	Medium

Table 6.1: Tempo and octave categories for pieces composed for each island

5 sets of viewpoints were considered to be used in order to create the lookup table, with viewpoints taken from the full list previously described in Chapter 5 in Table 5.2. The sets of viewpoints considered for this study are shown below:

***Conklin and Witten* (*1995*)**

These viewpoints are taken from Conklin and Witten (1995). Any viewpoints that could not be implemented (due to knowledge of bar structure or phrasal structure of the particular pieces, which is not available during a generated transition) were excluded, meaning that the final set of viewpoints were as follows:

- `ScaleDegree` $\otimes$ `Interval`
- `Interval` $\otimes$ `IOI`
- `Pitch`

M. Pearce* (*2005*) *System D

These viewpoints are taken from M. Pearce (2005)'s System D. Any viewpoints that could not be implemented (due to knowledge of bar structure or phrasal structure, for example) were excluded, meaning that the final set of viewpoints were as follows:

- `Interval` $\otimes$ `Duration`
- `ScaleDegree` $\otimes$ `IntervalFirstInSource`
- `Pitch` $\otimes$ `Duration`
- `ScaleDegree` $\otimes$ `Duration`
- `Interval` $\otimes$ `DurationRatio`

Highest Weighted

The entire list of viewpoints is sorted in descending order based on the entropy value for each viewpoint. This viewpoint set is made up of the 5 highest weighted viewpoints for the chosen pair of islands.

Mid Weighted

The entire list of viewpoints is sorted in descending order based on the entropy value for each viewpoint. This viewpoint set is made up the 5 mid weighted viewpoints for the chosen pair of islands.

Lowest Weighted

The entire list of viewpoints is sorted in descending order based on the entropy value for each viewpoint. This viewpoint set is made up the 5 lowest weighted viewpoints for the chosen pair of islands.

A test was run with 2 participants in order to populate the lookup table. Each participant was asked to watch 5 videos, with each video containing 9 different transitions between pieces of music from one island to another. Each transition was generated by using one of the viewpoint systems discussed above. Each participant was asked to rate the generated transition on a scale of 0 to 2, where:

- **0**: the transition was unsuccessful
- **1**: the transition needed some work
- **2**: the transition was largely successful

The island pairings presented to these participants were selected in such a way that each possible pairing of tempo and octave categories (as shown in Table 6.1) could be tested; this can be seen in Table 6.2 below.[10]

Tempo				**Octave**			
Source Category	**Target Category**	**Source Piece**	**Target Piece**	**Source Category**	**Target Category**	**Source Piece**	**Target Piece**
Fast	Fast	Forest 1	Carnival 1	High	High	Snow 1	Village 1
Fast	Medium	Carnival 2	Desert 2	High	Medium	Carnival 1	Desert 1
Fast	Slow	Carnival 1	Snow 1	High	Low	Forest 2	Spooky 2
Medium	Fast	Forest 2	Carnival 2	Medium	High	Desert 2	Snow 2
Medium	Medium	Desert 1	Village 2	Medium	Medium	Village 2	Carnival 2
Medium	Slow	Village 1	Spooky 1	Medium	Low	Carnival 2	Spooky 1
Slow	Fast	Spooky 1	Forest 1	Low	High	Spooky 1	Forest 1
Slow	Medium	Spooky 2	Snow 2	Low	Medium	Spooky 2	Desert 2
Slow	Slow	Snow 1	Spooky 2	Low	Low	*No such combination exists*	

Table 6.2: Island piece combinations used to create the lookup table

Table 6.3 shows an excerpt from the generated lookup table.

During the study, before a pair of islands are presented to participants, their corresponding source and target pieces are categorised according to their Tempo and Octave categories and referenced in the generated lookup table in order to select the best ranking viewpoint set. If, for example, the selected source piece was Forest 1, and the selected target piece was Ice 1, their respective Tempo and Octave categories are obtained and the total ratings for each viewpoint set compared, as can be seen in Table 6.4. In this example, the highest score obtained for the Tempo group was the Conklin and Witten (1995) viewpoint set with a score of 3, while the highest score obtained for the Octave group

[10]While both Spooky 1 and Spooky 2 are classified as Low, the environment does not allow for transition between the same island.

Octave: Conklin and Witten (1995)						
Source Category	**Target Category**	**Source Piece**	**Target Piece**	**Rating: Participant A**	**Rating: Participant B**	**Total Rating**
High	High	Ice 1	Village 1	**2**	**2**	**4**
High	Medium	Carnival 1	Desert 1	0	1	1
High	Low	Forest 2	Spooky 2	0	1	1
Medium	High	Desert 2	Ice 2	0	2	2
Medium	Medium	Village 2	Carnival 2	0	1	1
Medium	Low	Carnival 2	Spooky 1	1	2	3
Low	High	Spooky 1	Forest 1	1	1	2
Low	Medium	Spooky 2	Desert 2	1	2	3

Table 6.3: Excerpt from the generated lookup table, displaying the ratings given to transitions between pieces of different Octave categories using the *Conklin and Witten* (*1995*) viewpoint set

	Source Piece	**Target Piece**	**Total Rating**				
	Forest 1	Ice 1	**Conklin and Witten (1995)**	**M. Pearce (2005) System D**	**Highest Weighed**	**Mid Weighted**	**Lowest Weighted**
Tempo	Fast	Slow	**3**	1	2	2	2
Octave	High	High	**4**	**4**	0	0	1

Table 6.4: Finding the highest rated viewpoint set for two pieces. The highest rating for each category is marked in bold for convenience.

was either the Conklin and Witten (1995) viewpoint set or the M. Pearce (2005) System D viewpoint set with a score of 4. Therefore, either the Conklin and Witten (1995) viewpoint set or the M. Pearce (2005) System D viewpoint set would be used, chosen at random.

6.5 Methodology

Participants were taken to a quiet environment in order to take part in the study. This was conducted on a 64-bit Windows 10 laptop using the Intel HD Audio specification, and a pair of Sennheiser HD215 over-ear headphones were used in the study.

Participants were presented with an information sheet and a consent form, and given a brief explanation of what the study entailed.[11] They were also shown how to navigate the character through the 3D game environment, as well as given time to check if the volume levels were suitable.

Two "panic" keys were implemented just in case participants could not progress during the study due to the player character getting stuck in some way. F12 placed the character just before the bridge, rescuing characters that were stuck in the environment in some way and allowing players to still be able to cross the bridge and listen to the music transition. F7 skipped the particular island pair, allowing the participant to continue with the study; this was used a total of 3 times during the study due to error messages that prevented the music from continuing to play.

Each participant was assigned one of 16 study blocks, where each block was made up of both a custom transition or a crossfading transition between 15 pairs of islands. The order of the island pairs was randomised. For each island pair, participants were placed in one of three different locations on the island and were allowed to explore the immediate environment while listening to the corresponding music. After 10 seconds, the second island and the connecting bridge would appear, allowing participants to cross over to the second island by traversing the bridge. Doing so allows participants to listen to the transition (either a crossfade or the generated transition using the viewpoint system). When entering the second island, the first island and bridge are hidden, and the participant is allowed to explore the second island's environment for 8 seconds before being presented with an evaluation questionnaire, as shown in Fig. 6.12.

[11]The information leaflet given to participants for this study may be found in Appendix D

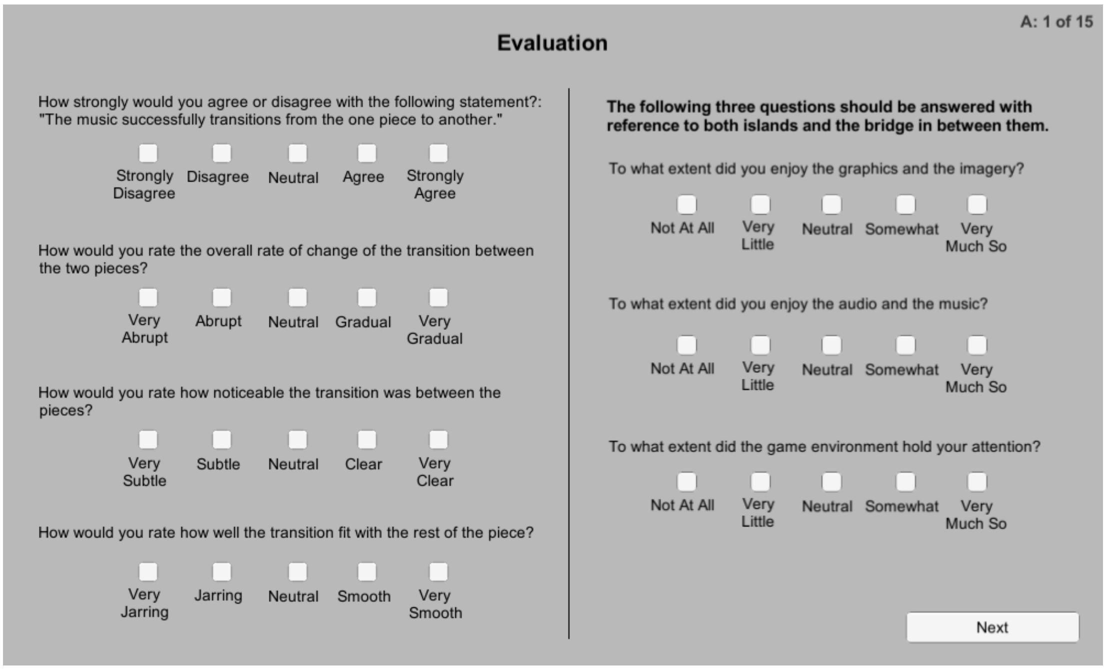


Figure 6.12: Questionnaire presented to participants after each pair of islands

6.5.1 Evaluating Transitions

As can be seen in Fig. 6.12, the left side of the questionnaire consisted of four different questions that pertain to evaluating the musical transition that the participant has experienced in the game environment. These questions are as follows:

- How strongly would you agree or disagree with the following statement?: "The music successfully transitions from the one piece to another."
    - Participants were given a 5-point scale with the following labels: *Strongly Disagree*, *Disagree*, *Neutral*, *Agree*, and *Strongly Agree*
- How would you rate the overall rate of change of the transition between the two pieces?
    - Participants were given a 5-point scale with the following labels: *Very Abrupt*, *Abrupt*, *Neutral*, *Gradual*, and *Very Gradual*

- How would you rate how noticeable the transition was between the pieces?
  - Participants were given a 5-point scale with the following labels: *Very Subtle*, *Subtle*, *Neutral*, *Clear*, and *Very Clear*
- How would you rate how well the transition fit with the rest of the piece?
  - Participants were given a 5-point scale with the following labels: *Very Jarring*, *Jarring*, *Neutral*, *Smooth*, and *Very Smooth*

These questions were similar to the ones used in the first study, as can be seen in Section 3.2.2 on page 87.

6.5.2 Immersion

One of the research questions that motivated the investigation of generative transitions in a video game is whether or not their use could increase the amount of immersion experienced by players while playing video games. The following questions presented to participants (found on the right side of Fig. 6.12) relate to level of immersion they experienced while interacting with the game environment in the given island pair.

- To what extent did you enjoy the graphics and the imagery?
  - Participants were given a 5-point scale with the following labels: *Not At All*, *Very Little*, *Neutral*, *Somewhat*, and *Very Much So*
- To what extent did you enjoy the audio and the music?
  - Participants were given a 5-point scale with the following labels: *Not At All*, *Very Little*, *Neutral*, *Somewhat*, and *Very Much So*
- To what extent did the game environment hold your attention?

– Participants were given a 5-point scale with the following labels: *Not At All*, *Very Little*, *Neutral*, *Somewhat*, and *Very Much So*

Nordin et al. (2014) presents an overview of different questionnaires that have been used in the literature to attempt to measure immersion. These were used as a reference in order to determine which questions should be used. In this study, three questions were selected to evaluate the immersion experienced by participants. Out of these three questions, the first two were taken from the questionnaire posed by Engström et al. (2015), itself a modified version of the *Immersive Experience Questionnaire* proposed by Jennett et al. (2008) and predominantly used in most studies pertaining to immersion in games. The third question was taken from the questionnaire found in Appendix B in Jennett et al. (2008).

These questions were chosen for several reasons. Most of the questions in the *Immersive Experience Questionnaire* were not relevant to the context (such as for example, questions about in-game challenges or game difficulty). Furthermore, the entire questionnaire could not be used due to its length, since it was intended to be used after a long session of gameplay rather than multiple times in the same session. Finally, the *Immersive Experience Questionnaire* does not contain any questions related to a player's auditory experience of the game, and hence the extra question added by Engström et al. (2015) was included, particularly since as discussed in Section 2.1.3.2, very little work has been conducted on which specific factors in music contribute towards immersion.

6.6 Conclusions

This chapter has presented a methodology aimed to evaluate the use of a new transition algorithm within the context of a video game environment, as compared to the crossfading transition technique. This evaluation is conducted based on four criteria: *Success*, *Rate of Change*, *Perception*, and *Degree of Fitting*, as well as the level of immersion experienced by participants. The following chapter analyses and discusses the results obtained from this study.

A performance of a composition which is indeterminate of its performance is necessarily unique. It cannot be repeated. When performed for a second time, the outcome is other than it was. Nothing therefore is accomplished by such a performance, since that performance cannot be grasped as an object in time. A recording of such a work has no more value than a postcard; it provides a knowledge of something that happened, whereas the action was a non-knowledge of something that had not yet happened.

John Cage, *Silence*, p. 39[1]

7

Evaluating Generative Transitions in Games

This chapter describes the techniques used to analyse the study described in Chapter 6. Section 7.1 describes the participants that took part in the study. Section 7.2 takes a look at the results of evaluating the transitions, while Section 7.3 investigates the results of evaluating immersion during transitions in a game environment.

[1]John Cage (1961). *Silence*. Wesleyan University Press. ISBN: 9780819560285, p. 39

7.1 Participants

Each participant was asked to fill in a questionnaire at the beginning of the study regarding their gaming habits and knowledge of music theory; this questionnaire is illustrated below in Fig. 7.1.

Figure 7.1: Questionnaire presented to participants before beginning the study

32 participants completed the study; split between 19 males, 12 females, and 1 person who actively identified as neither male nor female. The age of the participants ranged from 24-59 years old (mean age: 35.28, s.d. = 9.12). These participants were recruited via word-of-mouth as well as a call for participants on Twitter, and made up a convenience sample consisting of professors, lecturers, researchers, and students at the Open University.

Participants were grouped into 3 categories based on the number of times they reported they played video games during the week (Never, Less than 1 day a week, 1-2 days a week, 3-5 days a week, Every Day), with clarification given to people who asked what counted as a video game.[2] Participants were grouped as **Heavy Gamer**s if

[2]Here, the definition given to participants was the one presented by K. Collins on page 16: "any game consumed on video screens, whether these are computer monitors, mobile phones, handheld devices, televisions, or coin-operated arcade consoles" (K. Collins 2008, p. 3)

they reported as playing 3-5 days a week or every day, **Casual Gamer**s if they reported as playing less than 1 day a week or between 1-2 days a week, and **None** if they never play games. As illustrated below in Fig. 7.2, 22 participants were categorised as being Casual Gamers, 6 participants categorised as Heavy Gamers, and 9 participants as never playing games.

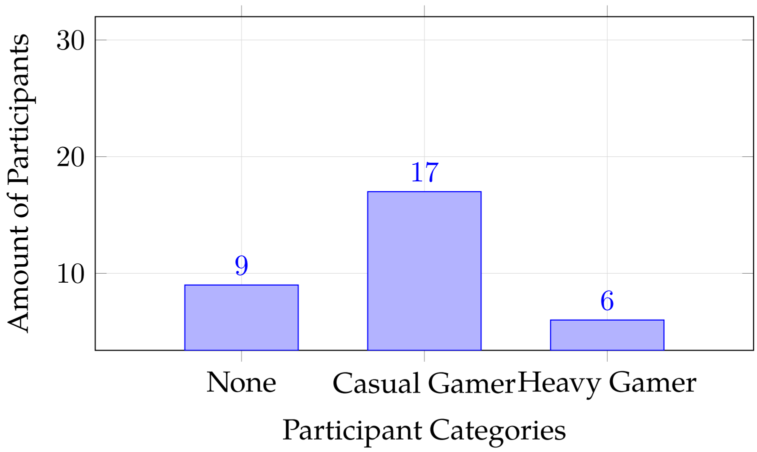


Figure 7.2: Participants grouped depending on how often they play video games

Similarly, participants were grouped into 3 categories based on their self-reported knowledge of music theory (None, Minimal, Some, Knowledgeable, Expert). Participants were grouped as having **Lots** of knowledge of music theory if they rated themselves as experts or knowledgeable, having **Some** amount of knowledge of music theory if they rated themselves as having some knowledge or minimal knowledge, and **None** if they rated themselves as having no knowledge of music theory. As illustrated below in Fig. 7.3, 17 participants were categorised as having Some knowledge of music theory, 5 participants categorised as having Lots of knowledge of music theory, and 5 participants as having no knowledge of music theory.

At the end of the study, participants were also given time to provide comments about their experience navigating the environment and about the study in general. Several participants stated that their perspective started to change while taking part in the study,

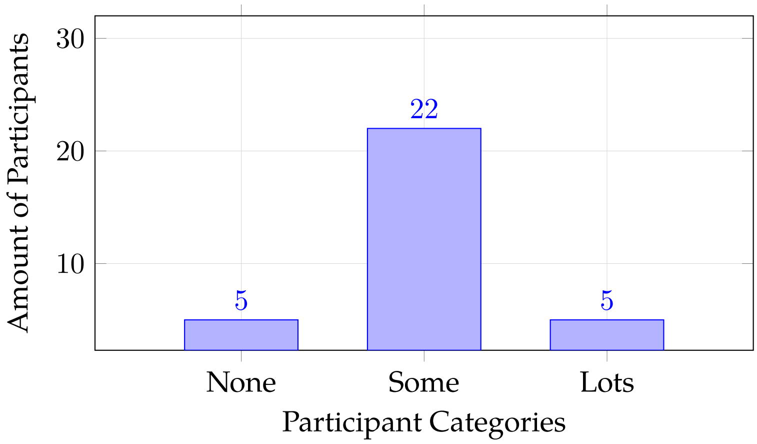


Figure 7.3: Participants grouped depending on their self-reported knowledge of music theory

where focus on the music increased as the novelty provided by the game environment decreased (and therefore, familiarity increased). Other participants stated that due to habituation, they started to recognise where to go, further contributing to this.

While one participant stated that they closed their eyes when the transition took place in order to completely focus on the music, another participant expressed difficulty in remembering what took place during the musical transition. A separate participant stated that their enjoyment of music was ranked low overall due to their expectation that the music would be similar to that of actual games (i.e. by using orchestral high-quality music and not by using melodies and MIDI instruments).

7.2 Evaluation of Transitions

The following subsections will take a look at the ratings given to transitions by each participant for each of the four categories: *Success*, *Rate of Change*, *Perception*, and *Degree of Fitting*.

7.2.1 Success

Fig. 7.4 shows the total ratings of Success given to each transition by each participant, split into the two techniques used: crossfading, and the custom algorithm used. As can be seen, the results seem to show that the custom algorithm is rated equally as successful as the crossfading algorithm.

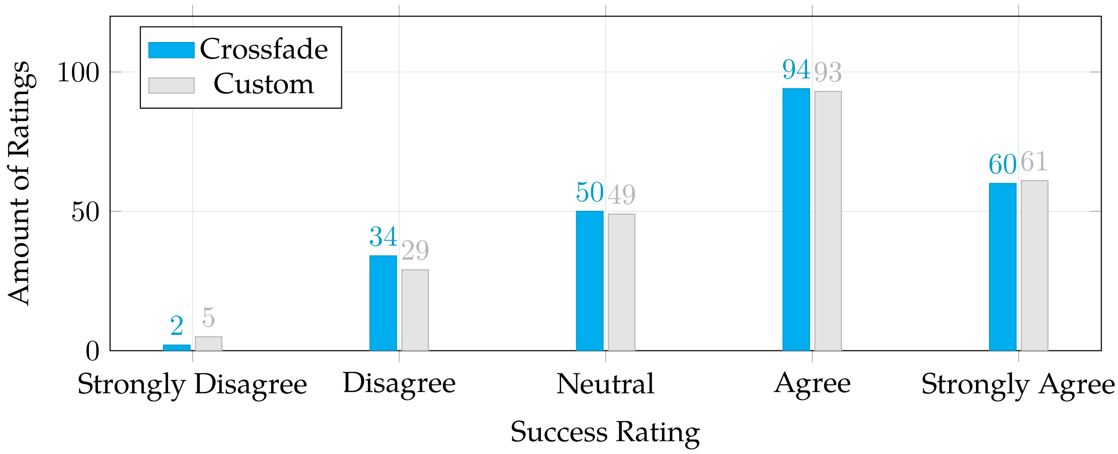


Figure 7.4: Ratings of transition success

The Kruskal-Wallis rank sum test shows that the averages of the distributions of success rankings for the crossfading and custom transitions are not significantly different, $\chi^2 = 0.039026$, df $= 1$, $p = 0.8434$. This means that for the rating of the transition's success, no difference was detected and crossfade and custom transition can be said to perform equally successfully in the context of a video game. Similarly, no differences were found in the success rankings of all transitions between gender categories ($\chi^2 = 1.3453$, df $= 2$, $p = 0.5104$).

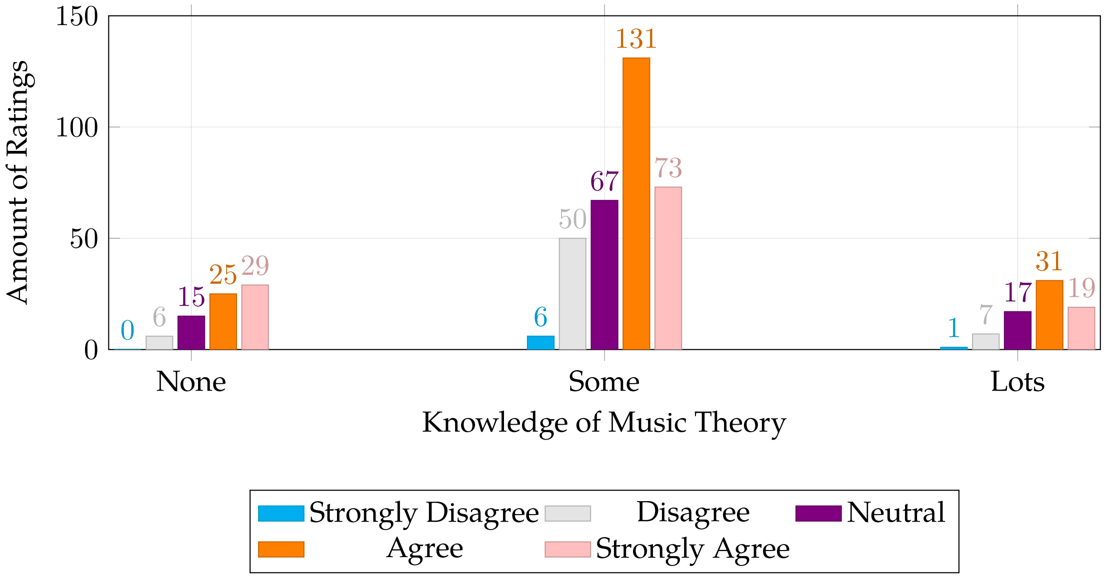


Figure 7.5: Difference in the rating of success between different categories of knowledge of music theory

However, the Kruskal-Wallis test indicates that the distributions produced by the groupings identified for a participant's knowledge of music theory ($\chi^2 = 7.959$, df $= 2$, $p = 0.01869$) came from different populations. This is illustrated in Fig. 7.5.

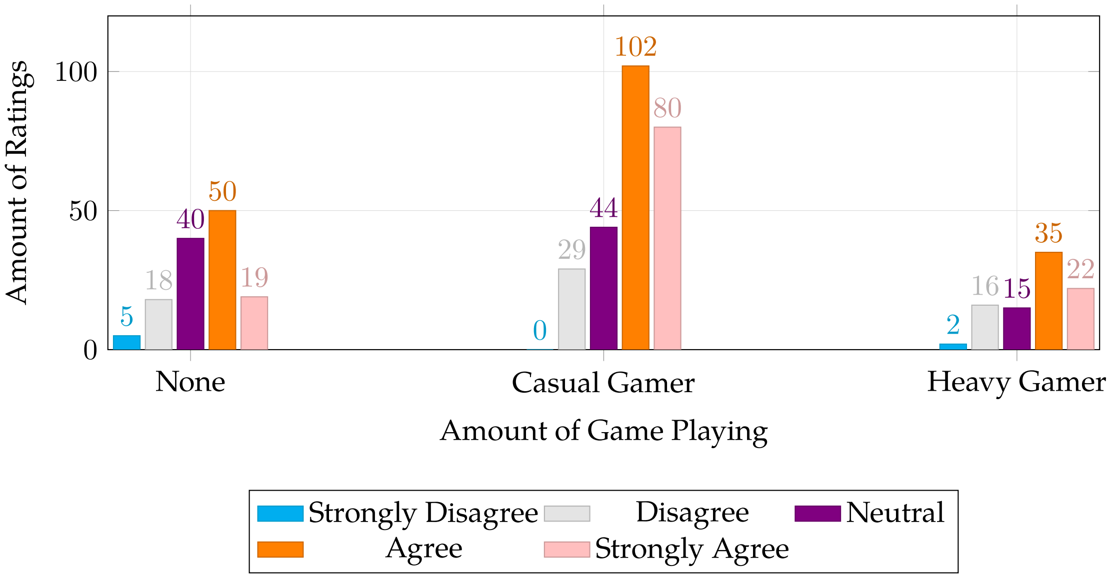


Figure 7.6: Difference in the rating of success between different categories of Gamer

Similarly, the Kruskal-Wallis test shows that the distributions produced by the groupings identified for a participant's amount of game playing ($\chi^2 = 18.264$, df $= 2$, $p = 0.0001081$, illustrated in Fig. 7.6), are from different populations altogether.

No differences were found in the success rankings of all transitions for different combinations of duration categories between source and target pieces (such as Fast to Fast, or Medium to Slow) ($\chi^2 = 12.007$, df $= 8$, $p = 0.1509$). However, significant differences were found in the success rankings of all transitions for different combinations of octave categories between source and target pieces (such as High to High, or Medium to Low) ($\chi^2 = 20.567$, df $= 7$, $p = 0.004467$).

Dunn's test was run as a post-hoc test in order to investigate which categories were different from each other by performing a pairwise comparison. The p-values were adjusted using the Benjamini-Hochberg method. For a participant's knowledge of music theory, this test resulted in only one combination of categories having different averages in their distributions, between None and Some ($p = 0.01603$). The rest of the combinations were not statistically significant (None and Lots: $p = 0.229$, Some and Lots: $p = 0.338$). For a participant's amount of game playing, this test resulted in the means of all 3 categories being different from each other (Casual Gamer to Heavy Gamer: $p = 9.5 \times 10^{-2}$, Casual Gamer to None: $p = 7.06 \times 10^{-5}$, and Heavy Gamer to None: $p = 9.8 \times 10^{-2}$.

The above results therefore show that the level of experience that participants have with games makes a difference as to how they would rate the transition's success, as can be seen in Fig. 7.6. Heavy gamers are more likely to rate a transition as being unsuccessful, while casual gamers are more likely to rate a transition as being overall successful.

7.2.2 Rate of Change

Fig. 7.7 shows the total ratings of Rate of Change given to each transition by each participant, split into the two techniques used: crossfading, and the custom algorithm used. As

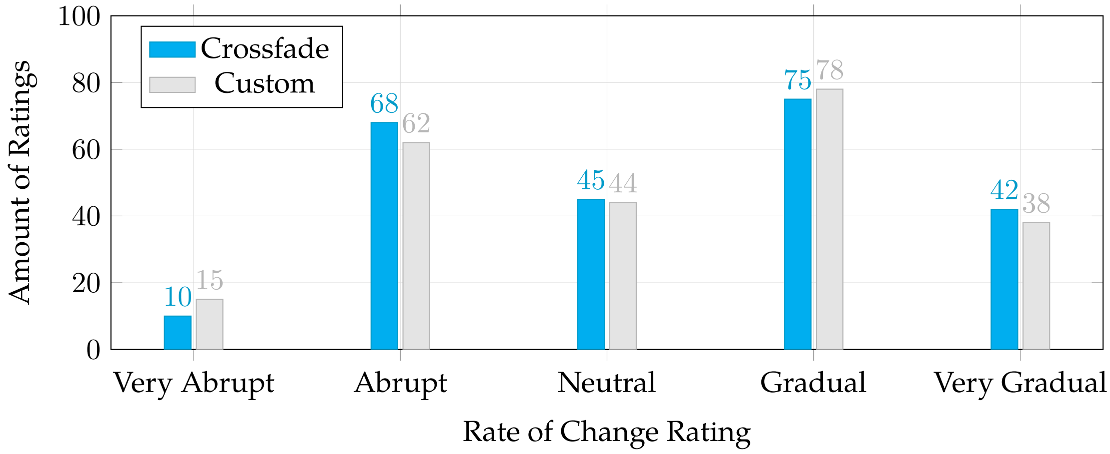


Figure 7.7: Ratings of transition rate of change

can be seen, the results seem to show that with regards to the transition's rate of change, the custom algorithm is rated as similar to the crossfading algorithm.

The Kruskal-Wallis rank sum test shows that the averages of the distributions of rate of change rankings for the crossfading and custom transitions are also not significantly different ($\chi^2 = 0.065999$, df $= 1$, $p = 0.7973$), This means that for the rating of the transition's rate of change, a crossfade and custom transition can be said to perform equally well in the context of a video game. Similarly, no differences were found in the rate of change rankings of all transitions between gender categories ($\chi^2 = 3.1473$, df $= 2$, $p = 0.2073$).

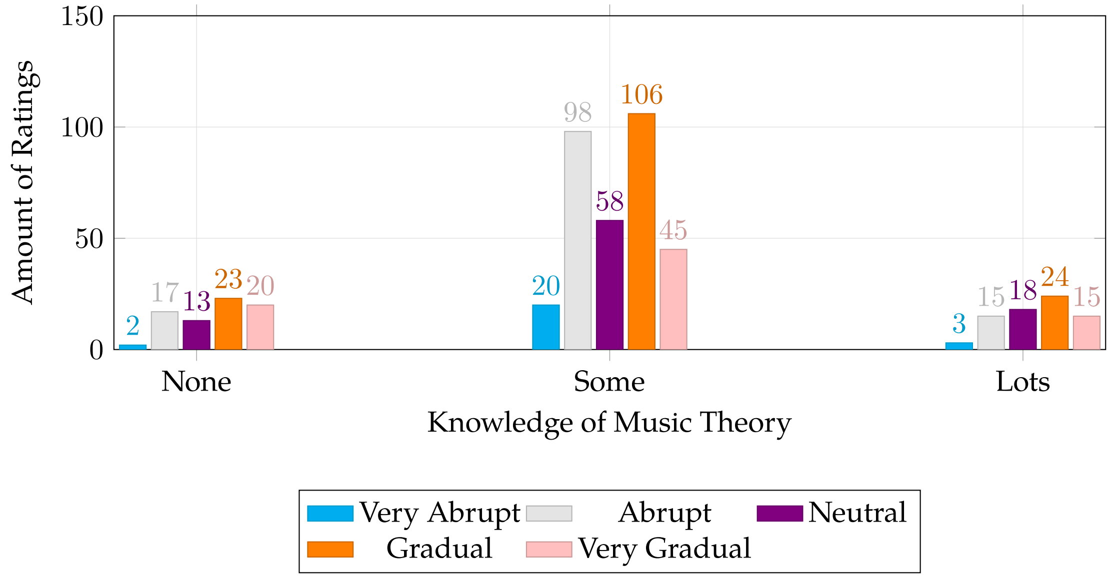


Figure 7.8: Difference in the rating of rate of change between different categories of knowledge of music theory

However, the Kruskal-Wallis test indicates that the distributions produced by the groupings identified for a participant's knowledge of music theory ($\chi^2 = 7.9346$, df $= 2$, $p = 0.01892$) are from different populations altogether. This is illustrated in Fig. 7.8.

Similarly, the Kruskal-Wallis test shows that the distribution produced by the groupings identified for a participant's amount of game playing ($\chi^2 = 6.3429$, df $= 2$, $p = 0.04194$, illustrated in Fig. 7.9), are also from different populations.

Significant differences were found in the rate of change rankings of all transitions for different combinations of duration categories between source and target pieces (such as Fast to Fast, or Medium to Slow) ($\chi^2 = 30.141$, df $= 8$, $p = 0.0001995$). However, no differences were found in the rate of change rankings of all transitions for different combinations of octave categories between source and target pieces (such as High to High, or Medium to Low) ($\chi^2 = 13.364$, df $= 7$, $p = 0.06372$).

Dunn's test was run as a post-hoc test in order to investigate which categories were different from each other by performing a pairwise comparison. The p-values were

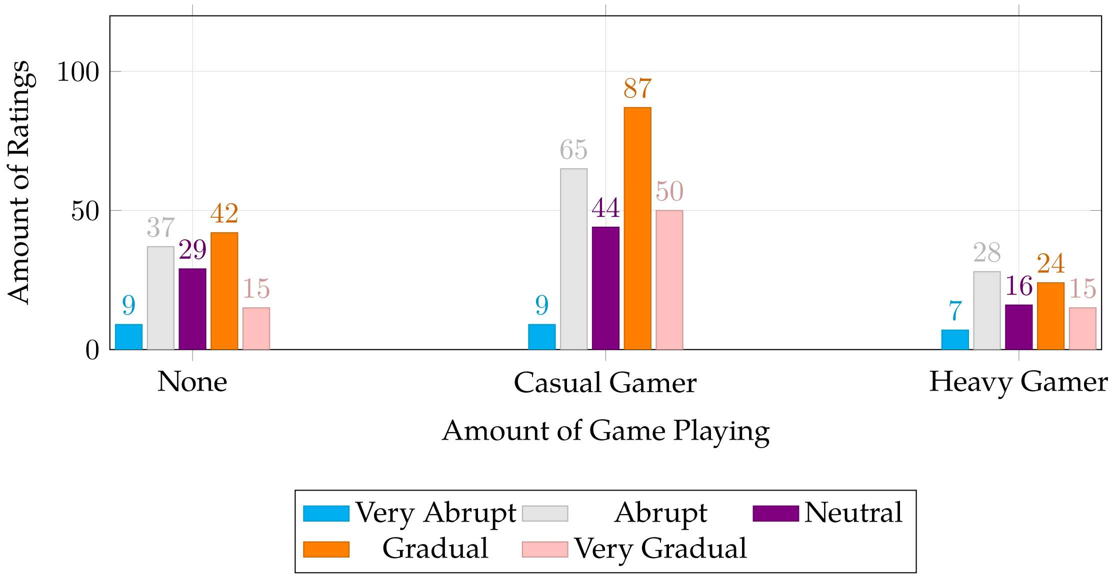


Figure 7.9: Difference in the rating of rate of change between different categories of Gamer

adjusted using the Benjamini-Hochberg method. For a participant's knowledge of music theory, this test resulted in only one combination of categories having different averages in their distributions, between None and Some ($p = 0.035$). The rest of the combinations were not statistically significant (None and Lots: $p = 0.526$, Some and Lots: $p = 0.131$). However, no differences were found between the categories established for the amount of experience participants had with playing games (None and Casual Gamer: $p = 0.0858$, None and Heavy Gamer: $p = 0.948$, Casual Gamer and Heavy Gamer: $p = 0.0983$).

7.2.3 Perception

Fig. 7.10 shows the total ratings of perception given to each transition by each participant, split into the two techniques used: crossfading, and the custom algorithm used. As can be seen, the results seem to show that with regards to the transition's perception, the custom algorithm is rated as similar to the crossfading algorithm.

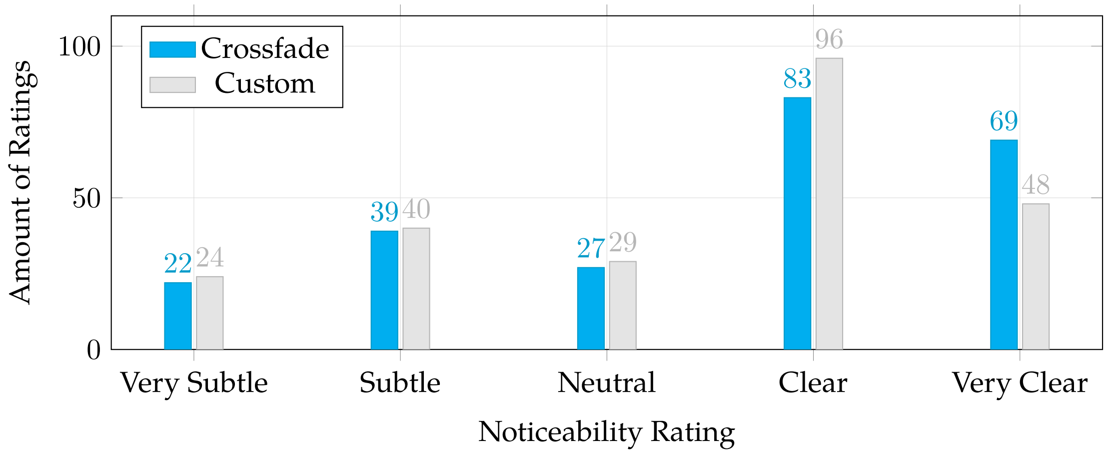


Figure 7.10: Ratings of transition perception

The Kruskal-Wallis rank sum test shows that the averages of the distributions of rankings for perception for the crossfading and custom transitions are not significantly different ($\chi^2 = 2.0231$, df $= 1$, $p = 0.1549$), This means that for the rating of the transition's perception, a crossfade and custom transition can be said to perform equally well in the context of a video game.

No significant differences were found in the averages of distributions for rankings for perception of all transitions between gender categories ($\chi^2 = 0.71984$, df $= 2$, $p = 0.6977$), a participant's knowledge of music theory ($\chi^2 = 0.033894$, df $= 2$, $p = 0.9832$) and for a participant's amount of game playing ($\chi^2 = 3.1648$, df $= 2$, $p = 0.2055$).

Significant differences were found in the perception rankings of all transitions for different combinations of duration categories between source and target pieces (such as Fast to Fast, or Medium to Slow) ($\chi^2 = 18.722$, df $= 8$, $p = 0.01642$), as well as in the perception rankings of all transitions for different combinations of octave categories between source and target pieces (such as High to High, or Medium to Low) ($\chi^2 = 15.903$, df $= 7$, $p = 0.02602$).

7.2.4 Degree of Fitting

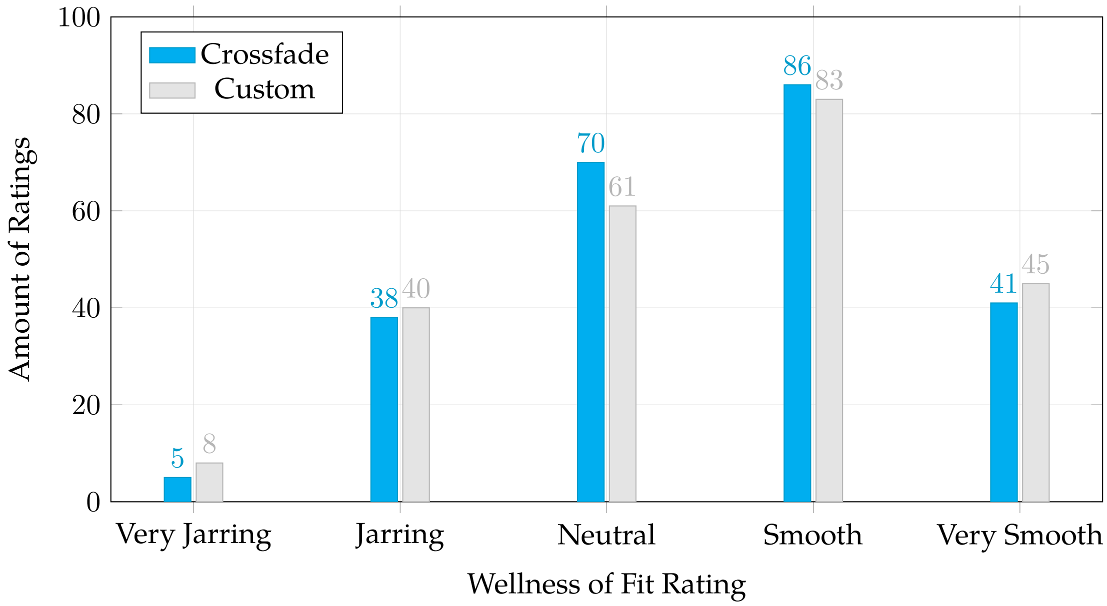


Figure 7.11: Ratings of transition degree of fitting

Fig. 7.11 shows the total ratings of Degree of Fitting given to each transition by each participant, split into the two techniques used: crossfading, and the custom algorithm used. As can be seen, the results seem to show that with regards to the transition's degree of fitting, the custom algorithm is rated as similar to the crossfading algorithm.

The Kruskal-Wallis rank sum test shows that the averages of the distributions of rankings for the degree of fitting for crossfading and custom transitions are also not significantly different ($\chi^2 = 0.007281$, df $= 1$, $p = 0.932$). This means that for the rating of the transition's degree of fitting, a crossfade and custom transition can be said to perform equally well in the context of a video game. Similarly, no significant differences were found in the degree of fitting rankings of all transitions between gender categories ($\chi^2 = 5.9907$, df $= 2$, $p = 0.05002$).

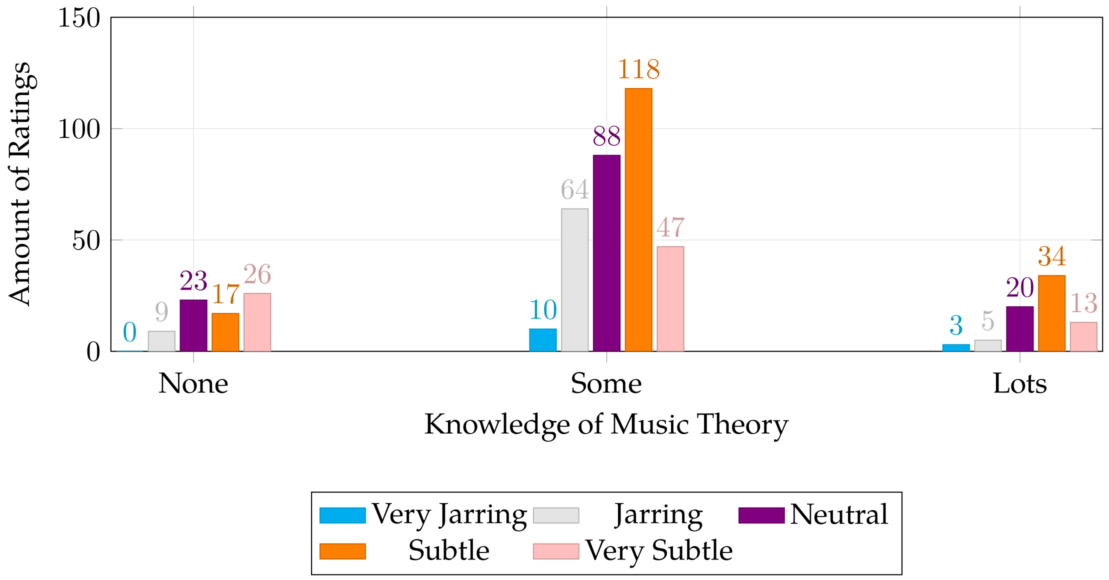


Figure 7.12: Difference in the rating of degree of fitting between different categories of knowledge of music theory

However, the Kruskal-Wallis test indicates that the distributions produced by the groupings identified for a participant's knowledge of music theory ($\chi^2 = 10.494$, df $= 2$, $p = 0.005263$) were from different populations altogether. This is illustrated in Fig. 7.12.

Similarly, the Kruskall-Wallis test showed that the distributions produced by the groupings identified for a participant's amount of game playing ($\chi^2 = 6.5502$, df $= 2$, $p = 0.03781$, illustrated in Fig. 7.13) are different populations altogether.

Significant differences were found in the degree of fitting rankings of all transitions for different combinations of duration categories between source and target pieces (such as Fast to Fast, or Medium to Slow) ($\chi^2 = 22.314$, df $= 8$, $p = 0.004366$), as well as in the degree of fitting rankings of all transitions for different combinations of octave categories between source and target pieces (such as High to High, or Medium to Low) ($\chi^2 = 18.333$, df $= 7$, $p = 0.01055$).

Dunn's test was run as a post-hoc test in order to investigate which categories were different from each other by performing a pairwise comparison. The p-values were

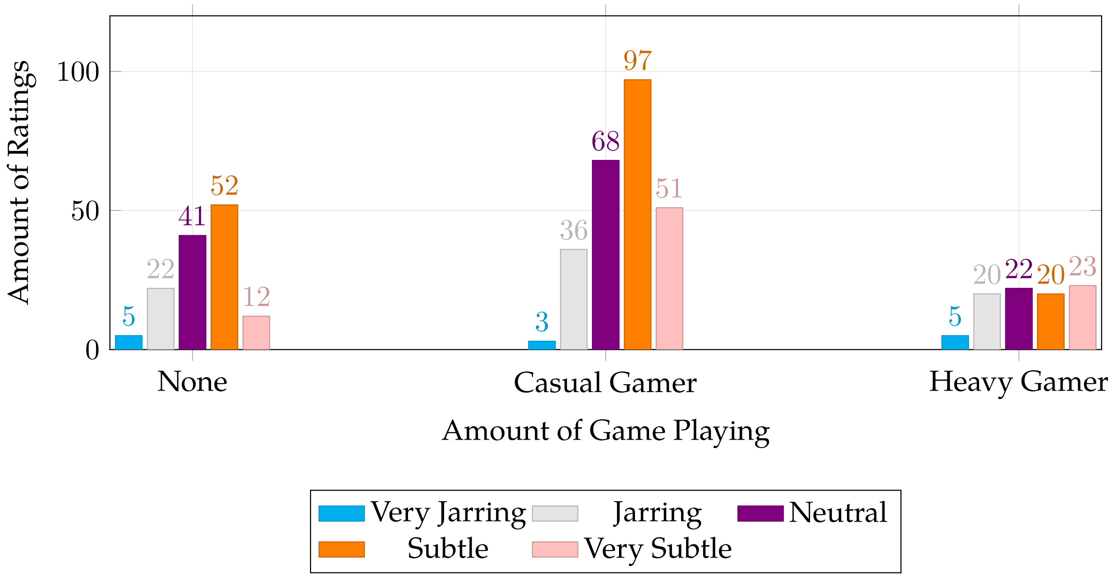


Figure 7.13: Difference in the rating of degree-of-fitting between different categories of Gamer

adjusted using the Benjamini-Hochberg method. For a participant's knowledge of music theory, this test resulted in only one combination of categories having different averages in their distributions, between None and Some ($p = 0.012$). The rest of the combinations were not statistically significant (None and Lots: $p = 0.502$, Some and Lots: $p = 0.0662$). Similarly, between the categories established for the amount of experience participants had with playing games, this test resulted in only one combination of categories having different averages in their distributions, between None and Casual Gamer ($p = 0.0428$) compared to None and Heavy Gamer ($p = 0.526$) and Casual Gamer and Heavy Gamer ($p = 0.226$).

7.3 Immersion

One main issue that participants expressed regarding the study is that due to the study conditions, they became used to the graphics and audio being presented, resulting in habituation. Several participants stated that, while at the beginning of the study they enjoyed spending their time exploring the islands presented and admiring the graphics and sounds, towards the end of the study, they became more focused on what was required of the study and less and less engaged with the environment itself. This meant that several participants expressed that they felt that answering the three questions pertaining to immersion was repetitive, again because they were engaging with the environment as a study and not as a game.

Other participants also stated that the questions themselves did not make sense due to the fact that an island pair was being presented; for example, one participant did not know how to answer if they enjoyed the audio of the first island, but did not enjoy the audio of the second island.

One possible reason for this is that immersion is normally experienced over a longer period of time (such as the 10 minutes given to participants by Grimshaw et al. (2008) for each study condition, or the 25 minutes given to participants by Engström et al. (2015), rather than the 1-2 minutes that each participant took for each island pair. Furthermore, while in both Grimshaw et al. (2008) and Engström et al. (2015) participants were presented with fully complete games that they could experience, here participants were presented with game environments where the only form of interaction they had was to look around and explore.[3]

The Kruskal-Wallis rank sum test shows that the averages of the distributions of ratings for total immersion for crossfading and custom transitions are not significantly different ($\chi^2 = 2.109$, df $= 1$, $p = 0.1464$). However, due to the issues stated above, this result is considered to be quite weak, and further research is necessary.

[3]The study took place in a room with sensor-activated lights that triggered when movement was detected. One participant was so engrossed with the study that the lights went out after some time and they didn't notice until much later. While an interesting anecdote, it is uncertain whether this level of focus was due to the game environment and the corresponding music, or due to the study conditions.

7.4 Conclusions

In conclusion, this study has shown that for the presented game environment, the custom transition algorithm was rated as performing equally as well as a crossfading transition across all four ratings (i.e. success, rate of change, perception, and degree of fitting). Since a crossfading transition is considered to be the industry standard in musical transitions, these results show that generative transitions can be a suitable alternative.

Some differences were also noticed in the ratings given between different participant categories. Heavy gamers were more likely to rate a musical transition as being unsuccessful compared to casual gamers; this might be because participants who have played more games have simply experienced more musical transitions and can more easily tell which ones work and which ones don't.

While the study also showed that the custom transition algorithm was ranked as equally as immersive as the crossfading transition, this result is taken to be inconclusive due to issues with transition length and evaluation as explained in Section 7.3.

But this didn't feel *like magic. It felt a lot older than that. It felt like music.*

Terry Pratchett, *Soul Music*[1]

8 Conclusions

In this final chapter, the main conclusions of this dissertation are discussed in relation to the original research question presented in Chapter 1, with several areas for future work being identified.

8.1 Research Insights

At the beginning of this dissertation, the following research question was presented:

[1] Terry Pratchett (1995). *Soul Music*. Paperback. Corgi

How can suitable musical transitions be generated automatically within the context of a digital game?

This question was further broken down into several sub-questions, each of which is examined further below.

8.1.1 Investigating the Detection of Musical Transitions

In Chapter 3, the following sub-questions were presented:

- How easily detectable are different transition techniques?
  - Is there a relationship between the evaluation criteria and how easily detectable transitions are?
  - Are people with musical experience better at detecting transitions?
- What are the criteria of success for a musical transition?
  - Are longer transitions more successful than shorter transitions?
  - Are transitions more effective between the same source and target piece, as opposed to different source and target pieces?

Each of these sub-questions are discussed in more detail below.

8.1.1.1 How easily detectable are different transition techniques?

Chapter 4 investigated how easily detectable four different transition techniques are: abrupt cut transition, horizontal resequencing, weighted averaging, and crossfading. As shown in Fig. 4.7, a significant amount of both abrupt cut transition and horizontal resequencing transitions were detected after 1.5 seconds. In contrast, just under 50% of crossfading transitions were detected after 7 seconds. This demonstrates that transitions that occur over a longer period of time (such as weighted averaging or crossfading) are harder to detect than transitions that occur instantly or over a shorter period of time.

As discussed in Section 4.2.3, evidence shows that there is a relationship between how easily detectable a transition is and the other evaluation criteria. A subtle transition is shown to change the music very gradually and have a high degree of fitting between the source and target pieces, making it more likely to be successful.

Finally, the last question asks whether people with more musical experience are better at detecting transitions. No significant difference was found between participants with different knowledge levels of music harmony, as well as their level of knowledge in playing an instrument. One interpretation of this result is that even with trained musicians or experts in music theory, a musical transition may be a more complex phenomenon than previously thought, particularly when considering the rate of detection for four different transition techniques (as discussed in Section 4.3.1).

8.1.1.2 What are the criteria of success for a musical transition?

While several features were listed in the literature as being required to have a successful transition, grouped into *rate of change* and *degree of fitting*, no evidence was found that these features had been experimentally tested. Based on the literature, the criteria of success for a musical transition were determined to be *rate of change*: that is, how quickly or slowly the source piece transitioned into the target piece, and *degree of fitting*: how well the musical transition fits between both the source and target pieces of music.

As discussed in Section 2.3, the literature states that longer transitions tend to be more successful due to their ability to bring the source music to a close and introduce elements of the target music. The results obtained in Chapter 4 show that this might not necessarily be the case, with shorter transition techniques such as the abrupt cut transition and horizontal resequencing technique being rated as more successful, more gradual, more subtle, and a better fit between the source and target pieces compared to longer transition techniques. This might indicate that shorter transition techniques may also have a place in the game composer's toolbox when used appropriately.

Finally, the effectiveness of transitions between the same two pieces was examined in Chapter 3. While only a small subset of the data, the results indicate that transitions between the same source and target piece were rated as more successful compared to transitions between different pieces. This result may indicate that generative transitions may be a suitable technique to use when looping music.

8.1.2 Generative Transitions in a Game Environment

In Chapter 6, the following sub-questions were presented:

- Can multiple viewpoint systems be used to generate musical transitions in a game environment?
  - Are multiple viewpoint systems suitable for use in real-time environments such as games?
  - How successful are these generated transitions?
  - Are some viewpoints better than others at generating musical transitions?
- Can generated transitions increase the amount of immersion players experience in games?

8.1.2.1 Can multiple viewpoint systems be used to generate musical transitions in a game environment?

A novel transition algorithm was created by using an implementation of the multiple viewpoint system (as previously described by Conklin (1990), M. Pearce (2005), Whorley (2013), and T. Hedges (2017)) and by using Markov models that were trained on a corpus of a single piece, the details of which are described in Chapter 5. This was implemented within a game environment, and a study (described in Chapters 6 and 7) was run in order to compare the new transition algorithm to the crossfading technique.

The implementation shown in this dissertation shows that the multiple viewpoint system is indeed suitable for use in real-time environments such as games. While the algorithm's integration in the game environment currently suffers from slight performance issues, the current implementation is fully functional and can certainly be optimised for use in a more complex scenario.

The transitions generated using the newly presented algorithm were also rated as comparable to the crossfading algorithm used as a benchmark, as can be seen in Fig. 7.4, implying that this technique is suitable for use in a video game environment. Since not all transition techniques are appropriate for all circumstances, this novel transition algorithm can be added to the composer's toolbox as another technique to be used when generating transitions. Furthermore, while already comparable to the crossfading algorithm, the technique presented in this dissertation has a lot of scope for future development

In this dissertation, different sets of viewpoints were used by using a pre-built lookup table to determine which set of viewpoints was the most appropriate to use to generate a transition. Since Conklin (1990) shows that the viewpoints used must be specific to the chosen corpus, no one particular set can be said to be better than another at generating musical transitions.

8.1.2.2 Can generated transitions increase the amount of immersion players experience in games?

Finally, the results of the new transition algorithm was evaluated based on the four previously discussed features (namely *success*, *rate of change*, *perception*, and *degree of fitting*), as well as questions taken from immersion questionnaires by Jennett et al. (2008)

and Engström et al. (2015). However, these questions may not have been suitable to use when focusing on measuring the amount of immersion that players have experienced due to the music, as discussed in Section 7.3. This may be because the questionnaire is designed for use over a longer experience with a more holistic focus, such as a 20-minute gameplay session, rather than short repeated sessions that is focused purely on music.

8.2 Future Work

This dissertation has introduced work that advances the field of research when discussing automatically generating musical transitions in a game environment. Several possibilities for future work based on this dissertation are given below.

8.2.1 Musical Transitions with Harmony

The work presented here focused on generating transitional melodies between two pre-composed melodies. While this dissertation did not take into account the effect of implied harmony that exists within a melody, future work may wish to take this into consideration in order to generate better melodic transitions.

The soundtrack for most modern games tends to be music of a more elaborate composition, such as by making use of harmony. While there has been some work done with automatically generating four part harmony (such as work done by Whorley (2013) and T. Hedges (2017) using multiple viewpoint systems), future work could focus on the use of generative harmony in transitions in a video game context by following in the footsteps of the aforementioned authors and applying it to an interactive context, as well as extending their work to be used in the generation of music with several melodic or harmonic lines.

8.2.2 Improvements to the Multiple Viewpoint System

There are several areas of future work with regards to the multiple viewpoint system. The dataset used for this dissertation was very small, and involved the use of one piece of music for the corpus of each viewpoint system. One immediate area for future work would be to collect and use a larger corpus that is suitable for generating musical transitions within the context of a video game.[2] This would allow the viewpoint system to have stronger predictive power and be able to generate better and more suitable transitions.

The current implementation of the multiple viewpoint system has followed similar approaches in the literature by hand-selecting a set of viewpoints beforehand in order to generate suitable transitions. Future work may involve developing a method of automatically selecting relevant viewpoints for generating a suitable transition. This will allow the multiple viewpoint system to be generalisable in order to be able to generate a transition between any pair of pieces.

While some viewpoints were added in order to cater for the generation of transitions, the vast majority of viewpoints used were based on previous work done by Conklin (1990), M. Pearce (2005), Whorley (2013), and T. Hedges (2017). One area of future work may involve the identification of specialised viewpoints which can be similarly generalisable, since as stated by T. Hedges (2017, p. 253): "[t]he potential for computationally learning the viewpoint representations themselves has been discussed as potential future work by M. Pearce (2005, pp. 220–221), however, to date this line of research has not been pursued". Other areas for future work may involve integrating a similarity metric between the source and target piece. This allows the transition boundary to be as wide or narrow as necessary to give the generated music enough time to transition successfully.

[2] For example, the datasets used by T. Hedges (2017, p. 91) each contain at least 150 pieces.

In this dissertation, Markov chains were used to model musical sequences in a multiple viewpoint system. As T. Hedges (2017, p. 66) states, "[a]lthough in theory any finite context model can be used, almost all multiple viewpoint systems to the author's knowledge use a form of Markov model". T. Hedges makes reference to one exception: Cherla et al. (2013)'s work on melodic prediction using Restricted Boltzmann Machines instead of Markov chains, which the authors claim helps to avoid data sparsity. One possible avenue for further work could be the exploration of alternative models for music generation to be used in a multiple viewpoint system, particularly in the context of musical transitions. Such examples may include the aforementioned Restricted Boltzmann Machines, or other models such as neural networks.

While the multiple viewpoint system has been used here to model musical events, they could be adapted in order to handle higher level structures such as motifs, phrases, or thematic content. This could be handled in a two-step process:

1. higher level viewpoints would first generate the probability of playing a new theme or a new phrase that needs to be played
2. if a new theme or phrase is needed, it is then populated by content from lower level viewpoints

Furthermore, the use of viewpoints may be modified to cater for structural events, such as the addition or removal of instruments during a piece. This would allow the *vertical reorchestration* transition technique (discussed in Section 2.3.3.5) to be modelled by the system.

8.2.3 Integration into a Game Engine

Hendrikx et al. (2013, p. 3) state that "[t]o avoid losing control over the design process, it is desirable that artists and designers can still influence the final product by adjusting the parameters of the procedure". This approach has been successfully implemented for visual assets such as in *SpeedTree* (Interactive Data Visualization Inc. 2002), where designers can tweak different parameters in order to get the model they require for their use-case.

However, this still seems to be an issue with generative music for games; for example, Somberg (2018) states that while there is some interest in the industry in real-time music, sound designers want a lot of control over the final product. This means that most of these tools tend to be used in an offline environment by the designer, rather than in the final product itself. Furthermore, Somberg states that since there aren't really any authoring tools that can be integrated cleanly into the developer environments that are currently being used, the use of these new techniques becomes difficult. Some progress has been made in this regard, with commercial products such as *Elias Studio* (Elias Software 2015) and *Melodrive* (Melodrive 2017) both offering algorithmic music libraries that can be integrated into popular game development environments.

Other developers have integrated algorithmic music systems into the game itself, allowing players to play around with the system. One such example is the platformer *Uurnog Uurnlimited* (Nifflas Games 2017).[3]

8.2.4 Evaluating Music Transitions

Most of the work conducted on evaluating algorithmic music revolves around the concept of *evaluating the output*; that is, the generated result of an algorithmic system. Such work includes:

- M. T. Pearce and G. A. Wiggins's (2007) focus on using expert judges to rate the stylistic success of generated pieces
- Ariza's (2009) discussion of musical Turing tests (such as the Lovelace test)

[3] A trailer featuring the algorithmic music generator can be seen here: `https://www.youtube.com/watch?v=WPOvohYeTKw`, with further improvements being made to the generator as can be seen here: `https://www.youtube.com/watch?v=n51Nmcqicr0`.

- Linson et al.'s (2012) distinction between qualitative and quantative evaluation techniques based on the context and use-case of the algorithmic system
- Jordanous's (2012) focus on evaluating systems based on the amount of creativity they express
- Prechtl et al.'s (2014b) distinction between player-focused and music-focused analysis techniques
- Williams et al.'s (2016) focus on evaluating player affect based on the generative music
- T. Collins and Laney's (2017) focus on evaluating generated music based on the corpus used to create it

However, one issue encountered in this research was to what extent these techniques could be used or adapted to evaluate the generated transitions, which may warrant future research.

This dissertation focused on rating transitions based on four different features: *success*, *rate of change*, *perception*, and *degree of fitting*. While these features were taken from the literature and an experimental investigation of these features carried out, future work could be conducted in this area in order to further refine these concepts. This is particularly important in light of the differences between evaluating the generation of entire pieces of music as opposed to evaluating transitions between two pieces.

One possible avenue for future work involves focusing on *evaluating the system*, rather than the output. Previous work has been done by Smith and Whitehead (2010) on analysing the expressive range of a generative system for game levels, as illustrated in Fig. 8.1 below. This is done by quantifying the resulting level based on a few chosen metrics, and plotting the results of multiple generated outputs in order to examiner potential biases that the algorithmic system contains.

Further work in this regard has been conducted by Cook et al. (2016) to incorporate such techniques into a Unity plugin that allows game developers to use such a tool to quickly select the best parameters for a level generator, as well as by Summerville (2018) in order to identify plagiarism and compare generated output to potential exemplars (such as the worst performing). Adapting such an approach to music with the appropriate metrics may give additional insight into the strengths and weaknesses of an algorithmic music system; initial work in this regard has been conducted by N. Collins (2008).

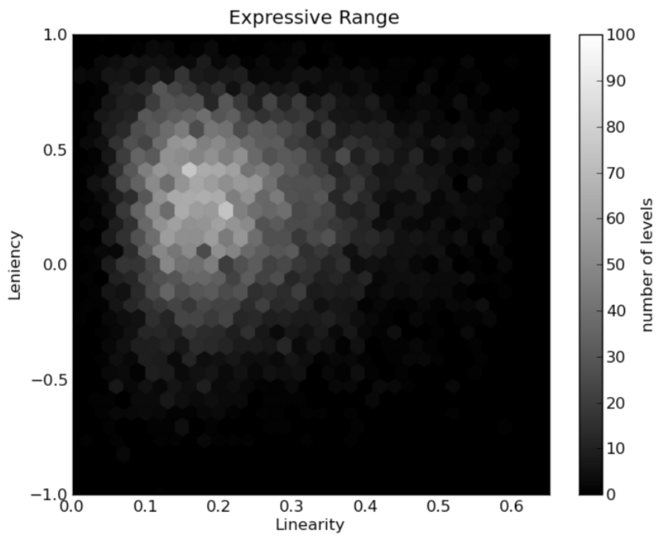


Figure 8.1: The expressive range of a level generator in terms of leniency and linearity. Reproduced from Smith and Whitehead (2010)

In a similar manner, Karth (2018) focuses on the poetics of generative systems by focusing on four different aspects of a generator: *complexity*, *form*, *locus*, and *variation*. While much of the paper discusses content such as game levels or environments (such as in *Minecraft* (Mojang 2011)) or algorithmically generated novels (such as Short (2015)'s *The Annals of the Parrigues*), future work may involve adapting Karth's frameworks for use in generative music systems.

Finally, Jan (2018, p. 2) states that "[the] human-centricity of music theory and analysis poses a problem for those wishing to extend it from [human-generated music] to [computer-generated music]", implying that techniques used to analyse music created by humans may need to be adapted or changed completely when dealing with the analysis of computer-generated music.

8.3 Closing Remarks

Throughout the years, video games have seen many advances such as high-quality real-time graphics and realistic physics among other areas. While the field of game music is still in its infancy, it has attracted a significant amount of interest over the last couple of years, with much of the focus being on dynamic music systems. This dissertation has shown that the area of musical transitions is still not yet fully understood, and that algorithmic techniques for music generation are a promising direction for future work in dynamic music in games.

9

References

A total of 285 references have been cited in this dissertation. These have been arranged into different sections depending on the publishing medium for the reader's convenience. Academic articles, books, websites, and other written material can be found under *Bibliography*, music can be found under *Discography*, games can be found under *Ludography*, software can be found under *Softography*, and videos can be found under *Videography*. The specific platform or version used for any games referenced in the *Ludography* section can be found in Chapter A.

All references cited are in English unless otherwise stated.

Bibliography (187 references cited)

Aarseth, Espen (2001). “Computer Game Studies, Year One”. In: *Game Studies* 1.1. URL: `http://gamestudies.org/0101/editorial.html` (cit. on p. 1).

Agresti, Alan (2007). *An Introduction to Categorical Data Analysis*. 2nd ed. John Wiley & Sons. ISBN: 9780471226185. DOI: `10.1002/0471249688` (cit. on p. 110).

Ames, Charles (1989). “The Markov Process as a Compositional Model: A Survey and Tutorial”. In: *Leonardo* 22.2, pp. 175–187 (cit. on p. 132).

Aristopoulos, Marios (2017). “A portfolio of recombinant compositions for the videogame Apotheon”. PhD. City, University of London, p. 208. URL: `http://openaccess.city.ac.uk/19298/` (cit. on p. 42).

Ariza, Christopher (2005). “Navigating the Landscape of Computer Aided Algorithmic Composition Systems: A Defintion, Seven Descriptors, and a Lexicon of Systems and Research”. In: *Proceedings of the International Computer Music Conference*. San Francisco, California, United States of America, pp. 765–772 (cit. on pp. 29, 32).

Ariza, Christopher (2009). “The Interrogator as Critic: The Turing Test and the Evaluation of Generative Music Systems”. In: *Computer Music Journal* 33.2, pp. 48–70. DOI: `10.1162/comj.2009.33.2.48` (cit. on p. 221).

Aspromallis, Christodoulos and Nicolas E. Gold (2016). “Form-Aware, Real-Time Adaptive Music Generation for Interactive Experiences”. In: *Proceedings of the Sound and Music Computing Conference 2016, SMC 2016*. Hamburg, Germany. URL: `http://discovery.ucl.ac.uk/1503636/` (cit. on pp. 3, 69, 70).

Atkinson, Sean E. (2019). "Soaring Through the Sky: Topics and Tropes in Video Game Music". In: *MTO: a Journal for the Society of Music Theory* 25.2. DOI: 10.30535/mto.25.2.1 (cit. on p. 186).

Barnes, Edward Shippen (n.d.). *Angels We Have Heard on High*. Mewsic Enterprises LLC. URL: http://www.christmascarolmusic.org/SATB/AngelsWeHaveHeard.html (cit. on p. 90).

Bateman, Chris (2016). "A Disavowal of Games". In: *Philosophical Perspectives on Play*. Ed. by Malcolm MacLean, Wendy Russell, and Emily Ryall. Routledge. Chap. 5, pp. 71–83 (cit. on pp. 13, 15).

Bech, Søren and Nick Zacharov (2006). *Perceptual audio evaluation - Theory, method and application*. Wiley-Blackwell, p. 462. ISBN: 0470869232 (cit. on pp. 81, 82).

van Beethoven, Ludwig (1975a). *Piano Sonata No.14, Op.27 No.2*. Ed. by Heinrich Schenker. With an intro. by Carl Schachter. New York, New York, United States of America (cit. on pp. xxix, 53, 54).

van Beethoven, Ludwig (1975b). *Piano Sonata No.5, Op.10 No.1*. Ed. by Heinrich Schenker. With an intro. by Carl Schachter. New York, New York, United States of America (cit. on pp. xxix, 52, 53).

Berger, Jonathan (1995). *Morphing Music at a High Level of Structure*. URL: https://ccrma.stanford.edu/%7B~%7Dbrg/research/morph/morph.html (cit. on p. 59).

Berndt, Axel (2009). "Musical Nonlinearity in Interactive Narrative Environments". In: *Proceedings of the International Computer Music Conference (ICMC 2009)*, pp. 355–358 (cit. on p. 3).

Berndt, Axel and Knut Hartmann (2008). "The Functions of Music in Interactive Media". In: *Proceedings of the First Joint International Conference on Interactive Digital Storytelling (ICIDS 2008)*, pp. 126–131. DOI: 10.1007/978-3-540-89454-4_19 (cit. on p. 18).

Berz, William L. and Anthony E. Kelly (1998). "Perceptions of More Complete Musical Compositions: An Exploratory Study". In: *Psychology of Music* 26.2, pp. 175–185. DOI: 10.1177/0305735698262005 (cit. on pp. 81, 82, 86).

Bessell, David (2002). "What's That Funny Noise? An Examination of the Role of Music in *Cool Boarders 2*, *Alien Trilogy* and *Medievil 2*". In: *ScreenPlay: Cinema/Videogames/Interfaces*. Ed. by Geoff King and Tanya Krzywinska. Wallflower Press. Chap. 8, pp. 136–144 (cit. on p. 20).

Beyls, Peter (1989). "The Musical Universe of Cellular Automata". In: *Proceedings of the 1989 International Computer Music Conference*. San Francisco, California, United States of America (cit. on p. 37).

Brill, Jonathan E. (2008). "Likert Scale". In: *Encyclopedia of Survey Research Methods: A-M*. Ed. by Paul J. Lavrakas. SAGE Publications Inc, pp. 427–429. DOI: 10.4135/9781412963947.n273 (cit. on p. 109).

Briot, Jean-Pierre, Gaëtan Hadjeres, and François-David Pachet (2019). *Deep Learning Techniques for Music Generation - A Survey*. arXiv: 1709.01620. URL: http://arxiv.org/abs/1709.01620 (cit. on p. 36).

Brooks, F. P., A. L. Hopkins, P. G. Neumann, and W. V. Wright (Sept. 1957). "An experiment in musical composition". In: *IRE Transactions on Electronic Computers* EC-6.3, pp. 175–182. DOI: 10.1109/TEC.1957.5222016 (cit. on p. 35).

Brown, Emily and Paul Cairns (2004). "A grounded investigation of game immersion". In: *Proceedings of CHI '04 – Extended Abstracts on Human Factors in Computing Systems*, pp. 1297–1300. ISBN: 1-58113-703-6. DOI: 10.1145/985921.986048 (cit. on p. 24).

Cage, John (1961). *Silence*. Wesleyan University Press. ISBN: 9780819560285 (cit. on p. 197).

Calleja, Gordon (2011). *In-Game: From Immersion to Incorporation*. MIT Press, p. 224. ISBN: 9780262015462 (cit. on pp. 24–26).

Chamberlain, Alan, Mads Bødker, Adrian Hazzard, David McGookin, David De Roure, Pip Willcox, and Konstantinos Papangelis (2017). "Audio Technology and Mobile Human Computer Interaction". In: *International Journal of Mobile Human Computer Interaction* 9.4, pp. 25–40. DOI: 10.4018/IJMHCI.2017100103 (cit. on pp. 72, 73).

Cherla, Srikanth, Tillman Weyde, Artur Garcez d'Avila, and Markus Pearce (2013). "A Distributed Model for Multiple-Viewpoint Melodic Prediction". In: *14th International Society for Music Information Retrieval Conference*. Curitiba, Paraná, Brazil (cit. on p. 220).

Chomsky, Noam (1957). *Syntactic Structures*. With an intro. by David W. Lightfoot. Mouton & Co. (cit. on p. 34).

Clarke, Eric F. and Carol L. Krumhansl (1990). "Perceiving Musical Time". In: 7.3, pp. 213–252. DOI: 10.2307/40285462 (cit. on pp. 81, 82, 86, 103).

Cohen, Jacob (1988). *Statistical Power Analysis for the Behavioral Sciences*. 2nd ed. Lawrence Erlbaum Associates, Publishers. ISBN: 9780805802832 (cit. on p. 110).

Collins, Karen (2007). "An Introduction to the Participatory and Non-Linear Aspects of Video Games Audio". In: *Essays on Sound and Vision*. Ed. by John Richardson and

Stan Hawkins. Helsinki, Finland: Helsinki University Press and Finnish Society for Ethnomusicology. Chap. 4, pp. 263–298 (cit. on pp. 20, 58).

Collins, Karen (2008). *Game Sound. An Introduction to the History, Theory, and Practice of Video Game Music and Sound Design*. MIT Press, p. 216. ISBN: 9780262033787 (cit. on pp. 6, 12, 16, 18, 27, 28, 74, 198).

Collins, Karen (2009). "An Introduction to Procedural Music in Video Games". In: *Contemporary Music Review* 28.1, pp. 5–15. DOI: 10.1080/07494460802663983 (cit. on pp. 2, 28, 30).

Collins, Karen (2013). *Playing with Sound: A Theory of Interacting with Sound and Music in Video Games*. ISBN: 9780262018678 (cit. on p. 18).

Collins, Nick (2008). "The Analysis of Generative Music Programs". In: *Organised Sound* 13.03, p. 237. DOI: 10.1017/S1355771808000332 (cit. on p. 223).

Collins, Nick (2018). "Origins of Algorithmic Thinking in Music". In: *The Oxford Handbook of Algorithmic Music*. Ed. by Roger T. Dean and Alex McLean. Oxford University Press. Chap. 4. DOI: 10.1093/oxfordhb/9780190226992.013.2 (cit. on p. 30).

Collins, Tom (2011). "Improved methods for pattern discovery in music, with applications in automated stylistic composition". PhD thesis. The Open University. URL: http://oro.open.ac.uk/30103/ (cit. on p. 35).

Collins, Tom and Robin Laney (2017). "Computer-generated stylistic compositions with long-term repetitive and phrasal structure". In: *Journal of Creative Music Systems* 1.2. DOI: 10.5920/JCMS.2017.02 (cit. on p. 222).

Conklin, Darrell (1990). "Prediction and Entropy of Music". MSc. University of Calgary (cit. on pp. 133–135, 216, 217, 219).

Conklin, Darrell and Ian H. Witten (1995). "Multiple viewpoint systems for music prediction". In: *Journal of New Music Research* 24.1, pp. 51–73. DOI: 10.1080/09298219508570672 (cit. on pp. 135, 144, 148–150, 188, 190, 191).

Cook, Michael, Jeremy Gow, and Simon Colton (2016). "Danesh: Helping bridge the gap between procedural generators and their output". In: *Proceedings of the 7th Workshop on Procedural Content Generation*, pp. 1–16 (cit. on p. 223).

Crawford, Chris (1984). *The Art of Computer Game Design: Reflections of a Master Game Designer*. Osborne/McGraw-Hill. ISBN: 0881341177 (cit. on p. 13).

Dannenberg, Roger B. (2006). "The Interpretation of MIDI Velocity". In: *Proceedings of the 2006 International Computer Music Conference*. New Orleans, Louisana, United States of America, pp. 193–196 (cit. on p. 95).

Davies, Matthew E. P., Philippe Hamel, Kazuyoshi Yoshii, and Masataka Goto (2014). "AutoMashUpper: Automatic creation of multi-song music mashups". In: *IEEE/ACM Transactions on Speech and Language Processing* 22.12, pp. 1726–1737. DOI: 10.1109/TASLP.2014.2347135 (cit. on p. 57).

Deliège, Irène (1987). "Grouping Conditions in Listening to Music: An Approach to Lerdahl & Jackendoff's Grouping Preference Rules". In: *Music Perception: An Interdisciplinary Journal* 4.4, pp. 325–359. ISSN: 07307829. DOI: 10.2307/40285378 (cit. on pp. 81, 103).

Eargle, John (2003). *Handbook of Recording Engineering*. 4th ed. Springer, p. 436. ISBN: 1402072309 (cit. on p. 64).

Ebcioğlu, Kemal (1988). "An expert system for harmonizing four-part chorales". In: *Computer Music Journal* 12.3, pp. 43–51. DOI: 10.2307/3680335 (cit. on p. 34).

Engström, Henrik, Jenny Brusk, and Per Anders Östblad (2015). "Including Visually Impaired Players in a Graphical Adventure Game: A Study of Immersion". In: *IADIS International Journal on Computer Science and Information Systems* 10.2, pp. 95–112 (cit. on pp. 25, 195, 211, 218).

Entertainment Retailers Association (2019). *Streaming drives entertainment sales 9.4% higher in 2018 to sixth consecutive year of growth but physical remains crucial to deliver megahits*. Entertainment Retailers Association. URL: https://eraltd.org/news-events/press-releases/2019/streaming-drives-entertainment-sales-94-higher-in-2018-to-sixth-consecutive-year-of-growth/ (visited on 01/22/2019) (cit. on p. 12).

Fernández, Jose David and Francisco Vico (2014). "AI Methods in Algorithmic Composition: A Comprehensive Survey". In: *Journal Of Artificial Intelligence Research* 48, pp. 513–582. DOI: 10.1613/jair.3908 (cit. on pp. 33, 35, 37, 133).

Freiknecht, Jonas and Wolfgang Effelsberg (Oct. 2017). "A Survey on the Procedural Generation of Virtual Worlds". In: *Multimodal Technologies and Interaction* 1.4, p. 27. DOI: 10.3390/mti1040027 (cit. on pp. 38, 40).

Fritsch, Melanie (2013). "History of video game music". In: *Music and Game: Perspectives on a popular alliance*. Ed. by Peter Moormann. Springer Fachmedien Wiesbaden. Chap. 1, pp. 11–41. ISBN: 978-3-531-17409-9. DOI: 10.1007/978-3-531-18913-0_1 (cit. on p. 27).

Garda, Maria B. (2013). "Neo-Rogue and the Essence of Roguelikeness". In: *Homo Ludens* 1.5, pp. 59–72 (cit. on p. 17).

General MIDI System Level 1 (1991). Tech. rep. Los Angeles, California, United States of America: The MIDI Manufacturers Association, p. 10 (cit. on pp. 95, 102).

Gibbons, William (2009). "Blip, Bloop, Bach? Some Uses of Classical Music on the Nintendo Entertainment System". In: *Music and the Moving Image* 2.1, pp. 2–14. JSTOR: 10.5406/musimoviimag.2.1.0040 (cit. on p. 91).

Gibbons, William (2011). "Wrap your troubles in dreams: Popular music, narrative, and dystopia in Bioshock". In: *Game Studies* 11.3 (cit. on p. 21).

Gillespie, Samuel and Oliver Bown (2017). "Solving adaptive game music transitions from a composer centred perspective". In: *5th International Workshop on Musical Metacreation (MUME 2017) at the Eighth International Conference on Computational Creativity (ICCC'17)*. Ed. by Philippe Pasquier, Oliver Bown, and Arne Eigenfeldt. Atlanta, Georgia, United States of America (cit. on p. 70).

Gluck, Michael (n.d.). *Final Fantasy - Prelude*. Piano Squall. URL: http://pianosquall.com/wp-content/uploads/2012/11/Final-Fantasy-Prelude1.pdf (cit. on p. 90).

Goldman, Richard Franko (1961). *Varèse: Ionisation; Density 21.5; Intégrales; Octandre; Hyperprism; Poème Electronique by Robert Craft and Varèse*. In: *The Musical Quarterly* vol. 47.1, pp. 133–134 (cit. on p. 17).

Gormanley, Stephen (2013). "Audio immersion in games – a case study using an online game with background music and sound effects". In: *The Computer Games Journal* 2.2, pp. 103–124 (cit. on p. 25).

Grimshaw, Mark, Craig A. Lindley, and Lennart E. Nacke (2008). "Sound and immersion in the first-person shooter: mixed measurement of the player's sonic experience". In: *Games Computing and Creative Technologies: Conference Papers (Peer-Reviewed)*, pp. 119–124. DOI: 10.1016/j.intcom.2010.04.005 (cit. on pp. 25, 211).

Gungormusler, Alper, Natasa Paterson-Paulberg, and Mads Haahr (2015). "barelyMusician: An Adaptive Music Engine For Video Games". In: *Proceedings of the 56th International Conference on Audio for Games*. London, United Kingdom (cit. on p. 44).

Hansrajh, Keton (2018). *The Expanded DMCA Exemption for Video Game Preservation Grants a Small Victory Amidst the Seventh Triennial Rulemaking*. Electronic Frontier Foundation. URL: `https://www.eff.org/deeplinks/2018/11/expanded-dmca-exemption-video-game-preservation-grants-small-victory-amidst` (visited on 12/03/2018) (cit. on p. 5).

Haydn, Joseph (1797). *Deutschlandlied*. Typesetter Johan Schoone. Cantorion. URL: `http://cantorion.org/music/3804/Song-of-Germany-%7B%5C%%7D28Deutschlandlied%7B%5C%%7D29-Voice-Piano`. Arrangement released under a Creative Commons Attribution-ShareAlike 3.0 license. (Cit. on pp. 89, 90).

Hedges, Stephen A. (1978). "Dice Music in the Eighteenth Century". In: *Music & Letters* 59.2, pp. 180–187. DOI: `10.1093/ml/59.2.180` (cit. on p. 30).

Hedges, Thomas (2017). "Advances in Multiple Viewpoint Systems and Applications in Modelling by Higher Order Musical Structure". PhD. Queen Mary University of London, p. 335 (cit. on pp. 133–135, 142–144, 147, 158, 163, 216, 218–220).

Hendrikx, Mark, Sebastiaan Meijer, Joeri van der Velden, and Alexandru Iosup (2013). "Procedural content generation for games: A survey". In: *ACM Transactions on Multimedia Computing, Communications, and Applications* 9.1, pp. 1–24. DOI: `10.1145/2422956.2422957` (cit. on pp. 39, 40, 220).

Herremans, Dorien, Ching-Hua Chuan, and Elaine Chew (2017). "A functional taxonomy of music generation systems". In: *ACM Computing Surveys* 50.5. ISSN: 15577341. DOI: `10.1145/3108242` (cit. on p. 33).

Herzfeld, Gregor (2013). “Atmospheres at Play: Aesthetical Considerations of Game Music”. In: *Music and Game: Perspectives on a popular alliance*. Ed. by Peter Moormann. Springer Fachmedien Wiesbaden. Chap. 8, pp. 147–157. DOI: 10.1007/978-3-531-18913-0 (cit. on p. 2).

Hild, Hermann, Johannes Feulner, and Wolfram Menzel (1991). “HARMONET: A Neural Net for Harmonizing Chorales in the Style of J. S. Bach”. In: *Advances in Neural Information Processing Systems*. Ed. by John E. Moody, Stephen J. Hanson, and Richard P. Lippman. Vol. 4, pp. 267–274 (cit. on p. 36).

Hiller, Lejaren and Leonard Isaacson (1957). *Iliac Suite*. Algorithmic Composition. Urbana and Champaign, Illinois, United States of America (cit. on p. 30).

Hoeberechts, Maia, Jeff Shantz, and Michael Katchabaw (2014). “Delivering Interactive Experiences through the Emotional Adaptation of Automatically Composed Music”. In: *The Oxford Handbook of Interactive Audio*. Ed. by Karen Collins, Bill Kapralos, and Holly Tessler. Oxford University Press. Chap. 25. DOI: 10.1093/oxfordhb/9780199797226.013.025 (cit. on p. 44).

Holloway, Bryan and Lippold Haken (1992). “A Sinusoidal Synthesis Algorithm for Generating Transitions Between Notes”. In: *Proceedings of the International Computer Music Conference (ICMC 1992)*. San José, California, United States of America (cit. on p. 50).

Hooper, Giles (2016). “Sounding the Story: Music in Videogame Cutscenes”. In: *Emotion in Video Game Soundtracking*. Ed. by Duncan Williams and Newton Lee. Springer International Publishing. Chap. 10, pp. 115–141. DOI: https://doi.org/10.1007/978-3-319-72272-6_10 (cit. on pp. 18, 23, 26).

Hudson, Robert A. (n.d.). *God Rest Ye Merry Gentlemen*. IMSLP: Petrucci Music Library. IMSLP: PMLP655340#404686 (cit. on p. 90).

Hulme, Zander (2017). "Achieving Seamless Transitions With Imbricate Audio". In: *G|A|M|E: The Italian Journal of Game Studies* 6. URL: https://www.gamejournal.it/killing-off-the-crossfade-achieving-seamless-transitions-with-imbricate-audio/ (cit. on p. 69).

Huron, David (1998). *Humdrum User's Guide*. URL: http://humdrum.ccarh.org/ (cit. on pp. 94, 351).

Jamison, Brian (2009). *Gamemastering*. Rampant Platypus Press, p. 330. ISBN: 9781448675432 (cit. on p. 24).

Jan, Steven B. (2018). "The theory and analysis of computer-generated music: A case-study of Colossus". In: *Proceedings of the Third Conference on the Computer Simulation of Musical Creativity*. Dublin, Ireland (cit. on p. 223).

Jennett, Charlene, Anna L. Cox, Paul Cairns, Samira Dhoparee, Andrew Epps, Tim Tijs, and Alison Walton (2008). "Measuring and defining the experience of immersion in games". In: *International Journal of Human Computer Studies* 66.9, pp. 641–661. DOI: 10.1016/j.ijhcs.2008.04.004 (cit. on pp. 195, 217).

Jeriaska (2011). *Myths, Mavericks, And Music Of Red Dead Redemption*. Gamasutra.com. URL: https://www.gamasutra.com/view/news/127900/Myths%7B%5C_%7DMavericks%7B%5C_%7DAnd%7B%5C_%7DMusic%7B%5C_%7DOf%7B%5C_%7DRed%7B%5C_%7DDead%7B%5C_%7DRedemption.php (visited on 08/03/2019) (cit. on pp. 62, 65).

Jordanous, Anna (2012). "A Standardised Procedure for Evaluating Creative Systems: Computational Creativity Evaluation Based on What it is to be Creative". In: *Cognitive Computation* 4.3, pp. 246–279. doi: 10.1007/s12559-012-9156-1 (cit. on p. 222).

Jørgensen, Kristine (2007). "On Transdiegetic Sounds in Computer Games". In: *Northern Lights: Film & Media Studies Yearbook* 5.1, pp. 105–117. url: http://www.atypon-link.com/INT/doi/pdf/10.1386/nl.5.1.105%7B%5C_%7D1 (cit. on pp. 21, 279).

Jørgensen, Kristine (2008a). "Left in the dark: playing computer games with the sound turned off". In: *From Pac-Man to Pop Music: Interactive Audio in Games and New Media*. Ed. by Karen Collins. Ashgate. Chap. 11, pp. 163–176. doi: 10.4324/9781351217743-12 (cit. on pp. 42, 80).

Jørgensen, Kristine (2008b). "Researching Players to Understand the Game". In: *Proceedings of the [Player] Conference*. Ed. by Sara Mosberg Iversen. Copenhagen, Denmark, pp. 197–219 (cit. on p. 21).

Jørgensen, Kristine (2010). "Time for New Terminology? Diegetic and Non-Diegetic Sounds in Computer Games Revisited". In: *Game Sound Technology and Player Interaction*. Ed. by Mark Grimshaw. IGI Global. Chap. 5, pp. 78–97. doi: 10.4018/978-1-61692-828-5.ch005 (cit. on pp. 21, 22).

Juul, Jesper (2005). *Half-Real: Video Games Between Real Rules and Fictional Worlds*. MIT Press. isbn: 9780262101103 (cit. on pp. 4, 13, 14, 16).

Kaae, Jesper (2008). "Theoretical approaches to composing dynamic music for video games". In: *From Pac-Man to Pop Music: Interactive Audio in Games and New Media*. Ed. by Karen Collins. Ashgate. Chap. 5, pp. 75–91. doi: 10.4324/9781351217743-6 (cit. on p. 75).

Kania, Andrew (2011). "Definition". In: *The Routledge Companion to Philosophy and Music*. Ed. by Theodore Gracyk and Andrew Kania. Routledge. Chap. 1, pp. 3–13 (cit. on p. 17).

Karth, Isaac (2018). "Preliminary Poetics of Procedural Generation in Games". In: *Proceedings of the 2018 DiGRA International Conference: The Game is the Message*. Turin, Italy (cit. on p. 223).

Kassabian, Anahid and Freya Jarman (2016). "Game and Play in Music Video Games". In: *Ludomusicology: Approaches to Video Game Music*. Ed. by Michiel Kamp, Tim Summers, and Mark Sweeney. Equinox Publishing Ltd. Chap. 7, pp. 116–132 (cit. on p. 16).

Kirnberger, Johann Philipp (1767). *Der allezeit fertige Polonoisen- und Menuettenkomponist*. German. Berlin, Germany: George Ludewig Winter. IMSLP: `PMLP243537#374746` (cit. on p. 30).

Korb, Darren (2012). *Bastion: Original Sheet Music*. Supergiant Games. URL: `https://store.supergiantgames.com/products/bastion-original-sheet-music` (visited on 01/30/2018). ZIP file (cit. on p. 91).

Land, Michael Z. and Peter N. McConnell (1994). "Method and apparatus for dynamically composing music and sound effects using a computer entertainment system". Patent US5315057 (United States of America). LucasArts Entertainment Company (cit. on p. 67).

Langston, Peter S. (1986). "(201) 644-2332 Eedie & Eddie on the Wire, An Experiment in Music Generation". In: *Proceedings of the Usenix Summer'86 Conference*. 201 (cit. on p. 43).

Langston, Peter S. (1989). “Six techniques for algorithmic music composition”. In: *Proceedings of the 15th International Computer Music Conference*. Columbus, Ohio, United States of America (cit. on p. 43).

Lankoski, Petri (2012). “Computer Games and Emotions”. In: *The Philosophy of Computer Games*. Ed. by John Richard Sageng, Hallvard Fossheim, and Tarjei Mandt Larsen. Springer. Chap. 4. DOI: 10.1007/978-94-007-4249-9_4 (cit. on p. 18).

Lerdahl, Fred and Ray Jackendoff (1996). *A Generative Theory of Tonal Music*. New Edition. MIT Press, p. 368. ISBN: 026262107X (cit. on pp. 31, 34, 81).

Lindenmayer, Aristid (1968). “Mathematical models for cellular interactions in development. II. Simple and branching filaments with two-sided inputs”. In: *Journal of Theoretical Biology* 18.3, pp. 300–315. DOI: 10.1016/0022-5193(68)90080-5 (cit. on p. 37).

Linson, Adam, Chris Dobbyn, and Robin Laney (2012). “Critical issues in evaluating freely improvising interactive music systems”. In: *Proceedings of the Third International Conference on Computational Creativity*, pp. 145–149 (cit. on p. 222).

Lopes, Phil, Antonios Liapis, and Georgios N. Yannakakis (2015). “Targeting Horror via Level and Soundscape Generation”. In: *11th AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*. AAAI Publications, pp. 37–43 (cit. on p. 45).

Macpherson, Stewart (1930). *Form in Music*. New and Revised. Joseph Williams Ltd. (cit. on pp. 51–53, 74).

Markov, Andrey Andreyevich (1924). *Исчисление вероятностей*. Russian. 4th. First published in 1900. Moscow, Russia: Государственное Издательство (cit. on p. 35).

Maurer IV, John A. (1999). *A Brief History of Algorithmic Composition*. URL: https://ccrma.stanford.edu/%7B~%7Dblackrse/algorithm.html (visited on 12/29/2018) (cit. on p. 30).

McAlpine, Kenneth B., Matthew Bett, and James Scanlan (2009). "Approaches to creating real-time adaptive music in interactive entertainment: A musical perspective". In: *Proceedings of the 35th Audio Engineering Society International Conference*. London, United Kingdom: Audio Engineering Society, pp. 94–104 (cit. on pp. 4, 23).

McCormick, Peter Dodds (n.d.). *Advance Australia Fair*. Typesetter Johan Schoone. Cantorion. URL: http://cantorion.org/music/3789/Advance-Australia-Fair-Voice-Piano. Arrangement released under a Creative Commons Attribution-ShareAlike 3.0 license. (Cit. on p. 90).

McSherry, Corynne, Kit Walsh, Mitch Stoltz, and Parker Higgins (2015). *Victory for Users: Librarian of Congress Renews and Expands Protections for Fair Uses*. Electronic Frontier Foundation. URL: https://www.eff.org/deeplinks/2015/10/victory-users-librarian-congress-renews-and-expands-protections-fair-uses (visited on 04/21/2019) (cit. on p. 5).

Medina-Gray, Elizabeth (2016). "Modularity in Video Game Music". In: *Ludomusicology: Approaches to Video Game Music*. Ed. by Michiel Kamp, Tim Summers, and Mark Sweeney. Equinox Publishing Ltd. Chap. 4, pp. 53–72 (cit. on pp. 42, 75).

Menabrea, Luigi Federico (1843). "Sketch of the Analytical Engine invented by Charles Babbage". In: *Scientific Memoirs: Selected from the Transactions of Foreign Academies of science and Learned Societies and from Foreign Journals*. Ed. by Richard Taylor. Trans. and comm. by Ada Lovelace. Vol. 3. London, United Kingdom: Richard and John E. Taylor. Chap. 29, pp. 666–731 (cit. on p. 127).

Meyer, Leonard B. (1961). *Emotion and Meaning in Music*. University of Chicago Press, p. 320. ISBN: 978-0226521398 (cit. on p. 105).

Miranda, Eduardo Reck (2001). *Composing Music with Computers*. With a forew. by Daniel V. Oppenheim. Focal Press. ISBN: 0240515676 (cit. on pp. 47–49).

Mozart, Wolfgang Amadeus (1793). *Musikalisches Würfelspiel*. K.516f. Plate 48. Bonn, Germany: N. Simrock (cit. on p. 30).

Munday, Rod (2007). "Music in Video Games". In: *Music, Sound and Multimedia: From the Live to the Virtual*. Ed. by Jamie Sexton. Edinburgh University Press, pp. 51–67 (cit. on pp. 23, 26, 27).

Muriel, Daniel and Garry Crawford (2018). *Video Games as Culture. Considering the Role and Importance of Video Games in Contemporary Society*. 1st ed. Routledge. ISBN: 9781138655119 (cit. on p. 12).

Nacke, Lennart and Craig A. Lindley (2009). "Affective Ludology, Flow and Immersion in a First-Person Shooter: Measurement of Player Experience". In: *Loading...: The Journal of the Canadian Game Studies Association* 3.5 (cit. on p. 25).

Nakamura, Jeanne and Mihaly Csikszentmihalyi (2002). "The Concept of Flow". In: *Handbook of Positive Psychology*. Ed. by C. R. Snyder and Shane J. Lopez. Oxford University Press. Chap. 7, pp. 89–105 (cit. on p. 25).

Nierhaus, Gerhard (2008). *Algorithmic Composition: Paradigms of Automated Music Generation*. 1st ed. Springer, p. 300. ISBN: 9783211755396. DOI: `10.1007/978-3-211-75540-2` (cit. on pp. 33–38, 131).

Nordin, A. Imran, Alena Denisova, and Paul Cairns (2014). “Too Many Questionnaires: Measuring Player Experience Whilst Playing Digital Games”. In: *Seventh York Doctoral Symposium on Computer Science & Electronics*. October, pp. 69–75 (cit. on p. 195).

Oppenheim, Daniel V. (1995). “Demostrating MMorph: A System for Morphing Music in Real-Time”. In: *ICMC95 International Computer Music Conference*. Banff, Alberta, Canada: ICMA, p. 2 (cit. on p. 62).

Oppenheim, Daniel Vincent (1997). *Interactive system for compositional morphing of music in realtime*. Patent. United States of America (cit. on pp. 59, 60, 62).

Pachet, François and Pierre Roy (2011). “Markov constraints: Steerable generation of Markov sequences”. In: *Constraints* 16.2, pp. 148–172. DOI: 10.1007/s10601-010-9101-4 (cit. on p. 35).

Papadopoulos, George and Geraint Wiggins (1999). “AI Methods for Algorithmic Composition: A Survey, a Critical View and Future Prospects”. In: *AISB Symposium on Musical Creativity*. Edinburgh, United Kingdom, pp. 110–117 (cit. on pp. 33, 34, 37).

Paul, Leonard J. (2013). “Droppin’ Science: Video Game Audio Breakdown”. In: *Music and Game: Perspectives on a Popular Alliance*. Ed. by Peter Moormann. Springer Fachmedien Wiesbaden. Chap. 3, pp. 63–80. DOI: 10.1007/978-3-531-18913-0_3 (cit. on pp. 67, 68).

Pearce, Marcus T. (2005). “The Construction and Evaluation of Statistical Models of Melodic Structure in Music Perception and Composition”. PhD. City University, London, p. 267 (cit. on pp. 133, 134, 137, 139, 143, 147–150, 157, 163, 188, 191, 216, 219, 351).

Pearce, Marcus T and Geraint A Wiggins (2007). “Evaluating cognitive models of musical composition”. In: *Proceedings of the 4th International Joint Workshop on Computational Creativity*. Ed. by Amílcar Cardoso and Geraint A. Wiggins. London, United Kingdom: Goldsmiths, University of London, pp. 73–80 (cit. on p. 221).

Pearce, Marcus, Darrell Conklin, and Geraint Wiggins (2005). “Methods for Combining Statistical Models of Music”. In: *Computer Music Modeling and Retrieval, Second International Symposium, CMMR 2004* (Esbjerg, Denmark). Ed. by Uffe Kock Wiil. Vol. 3310. February. Springer-Verlag, pp. 295–312. ISBN: 3540244581. DOI: `10.1007/978-3-540-31807-1_22` (cit. on p. 94).

Phillips, Winifred (2014). *A Composer's Guide to Game Music*. MIT Press. ISBN: 9780262026642 (cit. on pp. 58, 76).

Pierce, John R. (Nov. 1950). “Science for Art's Sake”. In: *Astounding Science Fiction*. Ed. by John W. Campbell Jr., pp. 83–92 (cit. on p. 167).

Pinchbeck, Dan, David Anderson, Janet Delve, Getaneh Otemu, Antonio Ciuffreda, and Andreas Lange (2009). “Emulation as a strategy for the preservation of games: the KEEP project”. In: *DiGRA '09 - Proceedings of the 2009 DiGRA International Conference: Breaking New Ground: Innovation in Games, Play, Practice and Theory*. London, United Kingdom (cit. on pp. 4, 5).

Pinkerton, Richard C. (1956). “Information Theory and Melody”. In: *Scientific American* 194.2, pp. 77–87. DOI: `10.1038/scientificamerican0256-77` (cit. on p. 35).

Polansky, Larry (1992). “More on Morphological Mutation Functions: Recent Techniques and Developments”. In: *Proceedings of the International Computer Music Conference 1992*. San José, California, United States of America, pp. 57–60 (cit. on p. 59).

Pratchett, Terry (1995). *Soul Music*. Paperback. Corgi (cit. on p. 213).

Prechtl, Anthony, Robin Laney, Alistair Willis, and Robert Samuels (2014a). "Algorithmic Music As Intelligent Game Music". In: *Proceedings of the 50th Anniversary Convention of the AISB*. London, United Kingdom (cit. on pp. 44, 70).

Prechtl, Anthony, Robin Laney, Alistair Willis, and Robert Samuels (2014b). "Methodological Approaches to the Evaluation of Game Music Systems". In: *Proceedings of the 2014 Audio Mostly Conference*. Aalborg, Denmark. DOI: `10.1145/2636879.2636906` (cit. on p. 222).

Prusinkiewicz, Przemyslaw (1986). "Score generation with L-systems". In: *Proceedings of the 1986 International Computer Music Conference*, pp. 455–457 (cit. on p. 37).

R Core Team (2017). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing. Vienna, Austria. URL: `https://www.R-project.org/` (cit. on p. 110).

Rimbault, Edward F., ed. (1846). *Nursery Rhymes, with the Tunes to which they are still sung in the nurseries of England, obtained principally from oral tradition*. Cramer, Beale & Co., p. 79 (cit. on p. 131).

Roads, Curtis (2001). *Microsound*. MIT Press, p. 409. ISBN: 0262182157 (cit. on pp. 47–49, 77).

Rondeau, Michel (2009). *While Shepherds Watched Their Flocks By Night*. The Cathedral Brass. IMSLP: `WIMA.dfd6-WHIChr#177079`. Arrangement released under a Creative Commons Attribution 3.0 license. (Cit. on p. 90).

Sabaneev, Leonid (1935). *Music for the Films*. Trans. from the Russian by S. W. Pring. London, United Kingdom: Sir Isaac Pitman & Sons Ltd, p. 128 (cit. on pp. 18, 75).

Sanders, Timothy and Paul Cairns (2010). “Time perception, immersion and music in videogames”. In: *Proceedings of the 24th BCS Interaction Specialist Group Conference*, pp. 160–167 (cit. on p. 25).

Schoenberg, Arnold (1970). *Fundamentals of musical composition*. Ed. by Gerald Strang and Leonard Stein. Faber and Faber. ISBN: 9780571196586 (cit. on pp. 54, 55).

Scirea, Marco, Peter Eklund, Julian Togelius, and Sebastian Risi (2018). “Evolving in-game mood-expressive music with MetaCompose”. In: *Proceedings of the Audio Mostly 2018 on Sound in Immersion and Emotion*. Wrexham, Wales, United Kingdom. DOI: `10.1145/3243274.3243292` (cit. on p. 45).

Shannon, Claude E. (1948). “A mathematical theory of communication”. In: *The Bell System Technical Journal* 27.July 1928, pp. 379–423. DOI: `10.1145/584091.584093` (cit. on pp. 35, 148).

Shockne, Raney (2015). *Dragon Age: Inquisition - The Bard Songs*. BioWare. URL: `http://assets.dragonage.com/content/tavern_songs/TavernSongs_ENG.zip` (visited on 01/30/2018). ZIP file (cit. on p. 91).

Short, Emily (2015). *The Annals of the Parrigues*. Algorithmically Generated Novel. URL: `https://drive.google.com/file/d/0B97d5C256qbrOHFwSUhsZE4tU0k/view` (cit. on p. 223).

Slaney, Malcolm, Michele Covell, and Bud Lassiter (1996). “Automatic audio morphing”. In: *1996 IEEE International Conference on Acoustics, Speech, and Signal Processing Conference Proceedings*. Vol. 2, pp. 1001–1004. DOI: `10.1109/ICASSP.1996.543292` (cit. on pp. 59, 60).

Small, Christopher (1998). *Musicking: The Meanings of Performing and Listening*. Middletown, Connecticut, United States of America: Wesleyan University Press, p. 232. ISBN: 9780819522573 (cit. on p. 79).

Smith, Gillian (2015). "An Analog History of Procedural Content Generation". In: *Proceedings of the 2015 Conference on the Foundations of Digital Games (FDG 2015)*. Monterey, California, United States of America (cit. on p. 38).

Smith, Gillian and Jim Whitehead (2010). "Analyzing the expressive range of a level generator". In: *Proceedings of the 2010 Workshop on Procedural Content Generation in Games - PCGames '10*. New York, New York, USA: ACM Press, pp. 1–7. DOI: 10.1145/1814256.1814260 (cit. on pp. 222, 223).

Sporka, Adam J. and Jan Valta (2017). "Design and implementation of a non-linear symphonic soundtrack of a video game". In: *New Review of Hypermedia and Multimedia* 23.4, pp. 229–246. DOI: 10.1080/13614568.2017.1416682 (cit. on pp. 70, 71).

Square Enix (2000). *Final Fantasy VIII Piano Collection Sheet Music - ピアノコレクションズ ファイナルファンタジー VIII (ピアノ　ソロ)*. Japanese. Yamaha Music Media. ISBN: 4636657772 (cit. on p. 91).

Square Enix (2002). *Final Fantasy X Piano Collection Sheet Music - 上級者向 ピアノコレクションズ ファイナルファンタジーX ピアノソロ*. Japanese. Yamaha Music Media. ISBN: 4636254287 (cit. on p. 91).

Stockburger, Axel (2003). "The Game Environment from an Auditive Perspective". In: *Level Up: Digital Games Research Conference Proceedings*. Ed. by Marinka Copier and Joost Raessens. Utrecht, The Netherlands (cit. on p. 24).

Stockhausen, Karlheinz (1956). *Klavierstück XI*. Catalogue Number 7. Universal Edition (cit. on pp. 42, 68).

Strank, Willem (2013). "The Legacy of iMuse: Interactive Video Game Music in the 1990s". In: *Music and Game: Perspectives on a popular alliance*. Ed. by Peter Moormann. Springer Fachmedien Wiesbaden. Chap. 4, pp. 81–91. ISBN: 978-3-531-17409-9. DOI: 10.1007/978-3-531-18913-0_4 (cit. on p. 67).

Strauss, Johann (n.d.). *Radetzky-Marsch, zu Ehren des grossen Feldherrn*. Plate T. H. 10,996. Vienna, Austria: Haslinger. IMSLP: PMLP68787#322630 (cit. on pp. 91, 92, 136, 150).

Strawm, John Michael (1985). "Modeling Musical Transitions". PhD. Stanford University, p. 251. URL: https://ccrma.stanford.edu/files/papers/stanm26.pdf (cit. on pp. 49, 50).

Strohbach, Siegfried (1963). *The Drunken Sailor*. IMSLP: PMLP587835#364028. Arrangement released under a Performance Restricted Attribution-NonCommercial-NoDerivs 3.0 license. (Cit. on pp. xxx, 90, 145, 150, 151).

Suits, Bernard (1978). *The Grasshopper. Games, Life, and Utopia*. University of Toronto Press, p. 180. ISBN: 0802023010 (cit. on p. 13).

Summanen, Henrik (2004). "Creating Music for Live-Action Role-Play". In: *Beyond Role and Play: Tools, Toys and Theory for Harnessing the Imagination*. Ed. by Markus Montola and Jaakko Stenros. Espoo, Finland: Solmukohta 2004, Knutepunkt, Ropecon. Chap. 22, pp. 225–230 (cit. on p. 19).

Summers, Tim (2011). "Playing the Tune: Video Game Music, Gamers, and Genre". In: *ACT: Zeitschrift für Musik & Performance* 2 (cit. on p. 27).

Summers, Tim (2016). *Understanding Video Game Music*. With a forew. by James Hannigan. Cambridge University Press. ISBN: 9781107116870. DOI: 10.1017/CBO9781316337851 (cit. on pp. 6, 11, 19).

Summerville, Adam (2018). "Expanding Expressive Range: Evaluation Methodologies for Procedural Content Generation". In: *Proceedings of the Fourteenth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE 18)*. Edmonton, Alberta, Canada: AAAI Press, pp. 116–122 (cit. on p. 223).

Sweeney, Mark Richard (2014). "The Aesthetics of Videogame Music". PhD. University of Oxford (cit. on p. 20).

Sweet, Michael (2014). *Writing Interactive Music for Video Games: A Composer's Guide*. Addison Wesley, p. 512. ISBN: 0321961587 (cit. on pp. 19, 22, 23, 55, 56, 58, 62, 68, 74, 75, 126).

Tchaikovsky, Pyotr (1892). *The Nutcracker (suite), Op. 71a*. Moscow, Russia: P. Jurgenson. IMSLP: PMLP03607#57454 (cit. on pp. 91, 92).

Tessler, Holly (2008). "The new MTV? Electronic arts and 'playing' music". In: *From Pac-Man to Pop Music: Interactive Audio in Games and New Media*. Ed. by Karen Collins. Ashgate. Chap. 1, pp. 13–25. DOI: 10.4324/9781351217743-2 (cit. on p. 12).

The MIDI Assocation (2019). *The MIDI Manufacturers Association (MMA) and the Association of Music Electronics Industry (AMEI) announce MIDI 2.0™ Prototyping*. The MIDI Assocation. URL: https://www.midi.org/articles-old/the-midi-manufacturers-association-mma-and-the-association-of-music-electronics-industry-amei-announce-midi-2-0tm-prototyping (visited on 01/22/2019) (cit. on p. 28).

Togelius, Julian, Emil Kastbjerg, David Schedl, and Georgios N. Yannakakis (2011a). "What is procedural content generation?: Mario on the borderline". In: *Proceedings of the 2nd International Workshop on Procedural Content Generation in Games - PCGames '11*. Bordeaux, France. DOI: 10.1145/2000919.2000922 (cit. on p. 38).

Togelius, Julian, Georgios N. Yannakakis, Kenneth O. Stanley, and Cameron Browne (2011b). "Search-based procedural content generation: A taxonomy and survey". In: *IEEE Transactions on Computational Intelligence and AI in Games* 3.3, pp. 172–186. DOI: 10.1109/TCIAIG.2011.2148116 (cit. on pp. 38, 40, 41).

Toiviainen, Petri (2000). "Symbolic AI versus connectionism in music research". In: *Readings in Music and Artificial Intelligence*. Ed. by Eduardo Reck Miranda. Routledge. Chap. 4, pp. 47–68. DOI: 10.4324/9780203059746-9 (cit. on p. 36).

Uebersax, John S. (2006). *Likert Scales: Dispelling the Confusion*. Statistical Methods for Rater Agreement. URL: http://john-uebersax.com/stat/likert.htm (visited on 05/29/2018) (cit. on pp. 108, 109).

van Elferen, Isabella (2016). "Analysing Game Musical Immersion: The ALI Model". In: *Ludomusicology: Approaches to Video Game Music*. Ed. by Michiel Kamp, Tim Summers, and Mark Sweeney. Equinox Publishing Ltd. Chap. 3, pp. 32–52. ISBN: 9781781791981 (cit. on p. 26).

Wade, John F. (1751). *O Come, All Ye Faithful*. Mewsic Enterprises LLC. URL: http://www.christmascarolmusic.org/SATB/OComeAllYe.html (cit. on p. 90).

Whalen, Zach (2004). "Play along - An Approach to videogame music". In: *Game Studies* 4.1 (cit. on p. 24).

White, John D. (1984). *The Analysis of Music*. 2nd ed. The Scarecrow Press, Inc (cit. on p. 54).

Whorley, Raymond Peter (2013). "The Construction and Evaluation of Statistical Models of Melody and Harmony". PhD. Goldsmiths, University of London, p. 405 (cit. on pp. 35, 131, 133, 147, 216, 218, 219).

Williams, Duncan, Jamie Mears, Alexis Kirke, Eduardo Miranda, Ian Daly, Asad Malik, James Weaver, Faustina Hwang, and Slawomir Nasuto (2016). "A Perceptual and Affective Evaluation of an Affectively-Driven Engine for Video Game Soundtracking". In: *ACM Computers in Entertainment* 14.3 (cit. on p. 222).

Winters, Ben (2010). "The non-diegetic fallacy: Film, music, and narrative space". In: *Music and Letters* 91.2, pp. 224–244. DOI: 10.1093/ml/gcq019 (cit. on p. 21).

Wittgenstein, Ludwig (1986). *Philosophical Investigations*. Trans. by G. E. M. Anscombe. 3rd ed. Basil Blackwell Ltd. ISBN: 0631119000 (cit. on p. 13).

Wooller, René (2007). "Techniques for automated and interactive note sequence morphing of mainstream electronic music". PhD. Queensland University of Technology, p. 355 (cit. on p. 59).

Wooller, René and Andrew R. Brown (2005). "Investigating morphing algorithms for generative music". In: *Proceedings of Third Iteration: Third International Conference on Generative Systems in the Electronic Arts* (cit. on pp. 59–62, 155).

Wooller, René, Andrew R. Brown, Eduardo Miranda, Rodney Berry, and Joachim Diederich (2005). "A framework for comparison of process in algorithmic music systems". In: *Generative Arts Practice*. Ed. by David Burraston and Ernest Edmonds. Sydney, Australia, pp. 109–124 (cit. on pp. 29, 31, 32, 76).

Young, Robert W. (1939). "Terminology for Logarithmic Frequency Units". In: *The Journal of the Acoustical Society of America* 11.1, p. 134. DOI: 10.1121/1.1916017 (cit. on p. 94).

Yu, Jason M. (2017). *Breaking the Loop: A Look at the Cinematic Music of Breath of the Wild*. The Musical Narrative. URL: http://jasonyu.me/breaking-the-loop-botw/ (visited on 04/19/2018) (cit. on p. 69).

Zehnder, Sean M. and Scott D. Lipscomb (2006). "The Role of Music in Video Games". In: *Playing video games: Motives, Responses, and Consequences*. Ed. by Peter Vorderer and Jennings Bryant. Lawrence Erlbaum Associates Publishers. Chap. 17, pp. 241–258 (cit. on p. 21).

Discography (2 references cited)

King Gizzard & the Lizard Wizard (2016). *Nonagon Infinity*. Audio CD. ATO Records (cit. on p. 56).

Midnight Syndicate (2003). *Dungeons and Dragons*. Audio CD. Entity Productions (cit. on p. 24).

Ludography (83 references cited)

.theprodukkt (2004). *.kkrieger*. PC. Video Game. .theprodukkt (cit. on pp. 13, 41, 265).

2K Boston and 2K Australia (2011). *Bioshock*. PC, Mac, Xbox 360, PS3. Video Game. 2K Games (cit. on pp. 21, 266).

Adams, Tarn (2006). *Dwarf Fortress*. PC, Mac, Linux. Video Game. Bay 12 Games (cit. on pp. 39, 267).

Alientrap (2015). *Apotheon*. PC, Mac, Linux, PS4. Video Game. Alientrap (cit. on pp. 42, 265).

Alspach, Ted and Akihisa Okui (2014). *One Night Ultimate Werewolf*. Tabletop Game. Bézier Games (cit. on pp. 24, 259).

Atari (1972). *Pong*. Arcade Cabinet. Video Game. Atari (cit. on pp. 1, 27, 269).

Barone, Eric (2016). *Stardew Valley*. PC, Mac, Linux, PS4, Xbox One, Nintendo Switch, PS Vita, iOS, Android. Video Game. Chucklefish (cit. on pp. 174, 270).

Bethesda Game Studios (2006). *The Elder Scrolls IV: Oblivion*. PC, PS3, Xbox 360. Video Game. Bethesda Softworks & 2K Games (cit. on pp. 21, 270).

BioWare (2014). *Dragon Age: Inquisition*. PS, PS3, PS4, Xbox 360, Xbox One. Video Game. Electronic Arts (cit. on pp. 93, 267).

Blizzard Entertainment (2004). *World of Warcraft*. PC, Mac. Video Game. Blizzard Entertainment (cit. on pp. 22, 63, 65, 73, 74, 77, 176, 179, 271).

Braben, David and Ian Bell (1984). *Elite*. BBC Micro, Acorn Electron, Apple II, Amstrad CPC, Commodore 64, ZX Spectrum, MSX, Tatung Einstein, IBM PC compatible, Acorn Archimedes, Amiga, Atari ST, Nintendo Entertainment System. Video Game. Acornsoft (cit. on pp. 39, 267).

Brace Yourself Games (2015). *Crypt of the NecroDancer*. PC, Mac, Linux, PS4, PSVita, iOS. Video Game. Klei Entertainment (cit. on pp. 17, 266).

Brace Yourself Games (2019). *Cadence of Hyrule*. Nintendo Switch. Video Game. Nintendo (cit. on pp. 17, 266).

Capcom (1996). *Resident Evil*. PC, PlayStation, Sega Saturn, Nintendo DS. Video Game. Capcom (cit. on pp. 6, 269).

Capcom (2017). *Resident Evil 7: Biohazard*. PC, PS4, Xbox One, Nintendo Switch. Video Game. Capcom (cit. on pp. 26, 269).

Clover Studio (2006). *Ōkami*. PS2, Wii. Video Game (cit. on pp. 4, 268).

Crowther, William and Don Woods (1977). *Colossal Cave Adventure*. DEC PDP-10. Video Game (cit. on pp. 14, 266).

Cuzzillo, Gabe (2019). *Ape Out*. PC, Nintendo Switch. Video Game. Devolver Digital (cit. on pp. 44, 265).

Cyan (1993). *Myst*. PC, Mac, 3DO, PlayStation. Video Game. Brøderbund (cit. on pp. 173, 268).

Dennaton Games (2012). *Hotline Miami*. PC, Mac, Linux. Video Game. Devolver Digital (cit. on pp. 12, 268).

Epic Games (2017). *Fortnite: Battle Royale*. PC, Mac, PS4, Xbox One, iOS, Nintendo Switch, Android. Video Game. Epic Games (cit. on pp. 267, 277).

Epic Stars Inc. (2016). *Pixelfield*. iOS, Android. Video Game. Cheyenne, Wyoming, United States of America: Epic Stars Inc. (cit. on pp. 43, 269).

Firaxis Games (2005). *Civilization IV*. PC, Mac. Video Game. 2K Games (cit. on pp. 12, 27, 266).

Fitterer, Dylan (2008). *Audiosurf*. PC, Zune. Video Game. Invisible Handlebar (cit. on pp. 17, 265).

Game Freak (1996). *Pokémon Red*. GameBoy. Video Game. Nintendo (cit. on pp. 23, 63, 174, 269).

Gearbox Software (2009). *Borderlands*. PC, Mac, PS3, Xbox 360. Video Game. 2K Games (cit. on pp. 39, 266).

Giant Squid Studios (2016). *Abzû*. PC, PS4, Xbox One, Nintendo Switch. Video Game. 505 Games (cit. on pp. 169, 265).

Gygax, Gary and Dave Arneson (1974). *Dungeons and Dragons*. Tabletop Role-Playing Game. TSR, Wizards of the Coast (cit. on pp. 38, 267).

Harmonix (2005). *Guitar Hero*. PS2. Video Game. RedOctane (cit. on pp. 16, 268).

Harmonix (2017). *Dropmix*. Tabletop Game. Hasbro (cit. on pp. 71, 267).

Hello Games (2016). *No Man's Sky*. PC, PS4, XBox One. Video Game. Hello Games (cit. on pp. 43, 268).

Hoeberechts, Maia, D. Kerr, R. Pacheco, Chris Eineke, B. Wheeler, Jeffrey Shantz, Nick DiZazzo, J. Bindra, Michael Katchabaw, and M. Park (2012). *Pop Tunes*. Video Game (cit. on pp. 44, 269).

id Software (2016). *DOOM*. PC, PS4, Xbox One, Nintendo Switch. Video Game. Bethesda Softworks (cit. on pp. 26, 267, 278).

Indieszero (2005). *Electroplankton*. Nintendo DS, Nintendo DSi. Video Game. Nintendo (cit. on pp. 16, 267).

Infinity Ward (2003). *Call of Duty*. PC, Mac, N-Gage. Video Game. Activision (cit. on pp. 266, 278).

Ion Storm (2004). *Thief: Deadly Shadows*. PC, Xbox. Video Game. Eidos Interactive (cit. on pp. 18, 271).

Jetsoft (1983). *Cavelon*. Arcade Cabinet, Commodore 64, ZX Spectrum. Video Game. Ocean Software (cit. on pp. 64, 266).

Konami (1981). *Frogger*. Arcade Cabinet. Video Game. Sega (cit. on pp. 64, 267).

Konami (1997). *Dance Dance Revolution*. Arcade Cabinet. Video Game. Konami (cit. on pp. 17, 267).

Konami Computer Entertainment Tokyo (2003). *Silent Hill 3*. PC, PS2. Video Game. Konami (cit. on pp. 18, 269).

Level-5 (2002). *Dark Chronicle*. PS2. Video Game. This game was released as *Dark Cloud 2* in North America. Sony Computer Entertainment (cit. on pp. 176, 267).

LucasArts (1991). *Monkey Island 2: LeChuck's Revenge*. Amiga, FM Towns, Mac, MS-DOS. Video Game. LucasArts (cit. on pp. 6, 63, 67, 174, 268).

LucasArts (1997). *The Curse of Monkey Island*. PC, Mac. Video Game. LucasArts (cit. on pp. 176, 270).

LucasArts (1998). *Grim Fandango*. PC. Video Game. LucasArts (cit. on pp. 26, 268).

Lucasfilm Games (1984). *Ballblazer*. Amstrad CPC, Apple II, Atari 8-bit, Atari 5200, Atari 7800, Commodore 64, Famicom, MSX, ZX Spectrum. Video Game. Atari (cit. on pp. 43, 266).

Lucasfilm Games (1990). *The Secret of Monkey Island*. Amiga, Atari ST, CDTV, FM Towns, Classic Mac OS, MS-DOS, Sega CD, iOS, PC, Mac, PS3, Xbox 360. Video Game. Lucasfilm Games (cit. on pp. 28, 270).

Marsden, Steve and David Cooke (1984). *Technician Ted*. Amstrad CPC, ZX Spectrum. Video Game. Hewson Consultants (cit. on pp. 91, 270).

Maxis (2008). *Spore*. PC, Mac, iPhone. Video Game. Electronic Arts (cit. on pp. 43, 270).

Mojang (2011). *Minecraft*. PC, Mac, Linux, Android, iOS, Windows Phone, Xbox 360, Xbox One, PS3, PS4, PS Vita, Raspberry Pi, Universal Windows Platform, Wii U. Video Game. Mojang, Microsoft Studios, Sony Computer Entertainment (cit. on pp. 13, 39, 223, 268, 279).

Monolith Productions (2000). *No One Lives Forever*. PC, Mac, PS2. Video Game. Fox Interactive (cit. on pp. 5, 268).

Naughty Dog (2007). *Uncharted: Drake's Fortune*. PS3, PS4. Video Game. Sony Computer Entertainment (cit. on pp. 22, 271).

Nifflas Games (2017). *Uurnog Uurnlimited*. PC, Mac, Linux, Nintendo Switch. Video Game (cit. on pp. 221, 271).

Nintendo (1985). *Super Mario Bros.* NES, Family Computer Disk System, Game Boy Color, Game & Watch, Arcade, PC-8801, PlayChoice-10, SNES, X1. Video Game. Nintendo (cit. on pp. 270, 279).

Nintendo (2017). *The Legend of Zelda: Breath of the Wild*. Nintendo Switch, Wii U. Video Game. Nintendo (cit. on pp. 68, 270).

Nintendo EAD (1998). *The Legend of Zelda: The Ocarina of Time*. Nintendo 64, GameCube, iQue Player. Video Game. Nintendo (cit. on pp. 42, 179, 270).

Nintendo EAD (2001). *Pikmin*. Gamecube, Wii. Video Game. Nintendo (cit. on pp. 68, 269).

Polys Entertainment (1997). *Gran Turismo*. PlayStation. Video Game. Sony Computer Entertainment (cit. on pp. 6, 267).

Prechtl, Anthony (2015). *Escape Point*. Video Game. Self-Published. ORO: 44172 (cit. on pp. 45, 70, 267).

PUBG Corporation (2017). *PlayerUnknown's Battlegrounds*. PC, Android, iOS, Xbox One, PS4. Video Game. PUBG Corporation (cit. on pp. 269, 277).

Rare (1998). *Banjo-Kazooie*. Nintendo 64, XBox 360. Video Game. Nintendo (cit. on pp. 68, 175, 179, 266, 279).

Rockstar North (2013). *Grand Theft Auto V*. PC, PS3, PS4, Xbox 360, Xbox One. Video Game. Rockstar Games (cit. on pp. 12, 268).

Rockstar San Diego (2010). *Red Dead Redemption*. PS3, Xbox 360. Video Game. Rockstar Games (cit. on pp. 18, 62, 65).

Rocksteady Studios (2015). *Batman: Arkham Knight*. PC, PS4, Xbox One. Video Game. Warner Bros. Interactive Entertainment (cit. on pp. 63, 77, 266).

SCEE Studio London (2004). *SingStar*. PS2. Video Game. Sony Computer Entertainment Europe (cit. on pp. 16, 269).

SEDIC (1987). *Otocky*. Family Computer Disk System. Video Game. ASCII Corporation (cit. on pp. 16, 268).

Sierra On-Line (1992). *King's Quest 6*. PC, Mac, MS-DOS, Amiga. Video Game. Sierra On-Line (cit. on pp. 22, 268).

Square (1999). *Final Fantasy VIII*. PlayStation, PS3, PSP, PSVita, PC. Video Game. Square (cit. on pp. 92, 267).

Square (2001). *Final Fantasy X*. PS2. Video Game. Sony Computer Entertainment (cit. on pp. 23, 63, 66, 92, 173, 178, 267).

StudioMDHR (2017). *Cuphead*. PC, Mac, Xbox One. Video Game. StudioMDHR (cit. on pp. 26, 266).

Supergiant Games (2011). *Bastion*. PC, Mac, Linux, Xbox 360, Xbox One, iOS, PS4, PSVita. Video Game. Warner Bros. Interactive Entertainment (cit. on pp. 93, 266).

Superhot Team (2016). *Superhot*. PC, Mac, Linux, PS4, Xbox One. Video Game. Superhot Team (cit. on pp. 169, 270).

Taito (1978). *Space Invaders*. Arcade Cabinet. Video Game. Taito, Midway, Leisure & Allied Industries (cit. on pp. 27, 270).

Tale of Tales (2006). *The Endless Forest*. PC. Video Game. Tale of Tales (cit. on pp. 13, 270).

thatgamecompany (2012). *Journey*. PC, PS3, PS4. Video Game. Sony Computer Entertainment (cit. on pp. 12, 175, 178, 268).

Thekla Inc. (2016). *The Witness*. PC, Mac, PS4, Xbox One, iOS, Nvidia Shield. Video Game. Thekla Inc. (cit. on pp. 169, 178, 270).

Toy, Michael, Glenn Wichman, and Ken Arnold (1980). *Rogue*. Amiga, Amstrad CPC, Atari 8-bit, Atari ST, Commodore 64, DOS, Mac, TRS-80 CoCo, Unix, ZX Spectrum. Video Game (cit. on pp. 17, 39, 269).

Treasure (1994). *Dynamite Headdy*. Sega Genesis, Game Gear, Master System. Video Game. Sega (cit. on pp. 91, 267).

Turbine Entertainment Software (1999). *Asheron's Call*. PC. Video Game. Microsoft, Turbine, Warner Bros. Interactive Entertainment (cit. on pp. 5, 265).

United Game Artists (1999). *Space Channel 5*. Dreamcast, PS2, Game Boy Advance. Video Game. Sega (cit. on pp. 16, 269).

Upper One Games (2014). *Never Alone* (*Kisima Innjitchuŋa*). PC, Mac, Linux, PS4, PS4, Xbox One, iOS. Video Game. E-Line Media (cit. on pp. 175, 268).

Warhorse Studios (2018). *Kingdom Come: Deliverance*. PC, PS4, XBox One. Video Game. Deep Silver, Warhorse Studios (cit. on pp. 70, 71, 173, 268).

Westwood Studios (1992). *Dune II*. Amiga, MS-DOS, RISC OS, Genesis/Mega Drive. Video Game. Virgin Interactive (cit. on pp. 27, 267).

Wieslander, Eliot and Katarina Björk (2003). *Mellan himmel och hav*. Swedish. Live-Action Role-Playing Game (cit. on pp. 19, 268).

Softography (6 references cited)

Bézier Games (2013). *One Night*. iOS, Android. Accompanying app for the game *One Night Ultimate Werewolf* (Alspach and Okui 2014). Bézier Games (cit. on p. 24).

Edlund, Jonas (1999). *Musifier Adaptive Improviser*. Stockholm, Sweden: InterAmus Music Systems. URL: `http://musifier.com` (cit. on p. 59).

Elias Software (2015). *Elias Studio*. Stockholm, Sweden. URL: `http://www.eliassoftware.com` (cit. on pp. 45, 221).

Interactive Data Visualization Inc. (2002). *SpeedTree* (cit. on pp. 39, 221).

Loomes, Benjamin (2008). *Syrinscape*. Sydney, Australia. URL: `http://www.syrinscape.com` (cit. on p. 72).

Melodrive (2017). *Melodrive*. Berlin, Germany: Melodrive. URL: http://melodrive.com/ (cit. on p. 221).

Videography (7 references cited)

Barreau, Pierre (Oct. 2, 2018). *How AI could compose a personalized soundtrack to your life*. Vancouver, British Columbia, Canada: TED. URL: https://www.youtube.com/watch?v=wYb3Wimn01s (cit. on p. 43).

Disney, Walt and Ub Iwerks (1928). *Steamboat Willie*. United States of America: Celebrity Productions, Cinephone (cit. on pp. 18, 19).

Frey, Brad (2019). *Composing Cinematic Music Quickly w/ AIVA - "A Diary of Winter" Making Of*. URL: https://www.youtube.com/watch?v=msuue06wvVU (cit. on p. 43).

Jolly, Kent and Aaron McLeran (2008). *Procedural Music in Spore*. San Francisco, California, United States of America: Game Developers' Conference. URL: https://www.gdcvault.com/play/323/Procedural-Music-in (cit. on p. 43).

Page, Jason and Michael Kelly (2007). *PS3 Audio: More Than Extra Channels*. San Francisco, California, United States of America: Game Developers' Conference. URL: http://www.gdcvault.com/play/649/PS3-Audio-More-Than-Extra (cit. on p. 28).

Somberg, Guy (Nov. 21, 2018). *Game audio programming*. London, United Kingdom: Audio Developer Conference 2018. URL: https://www.youtube.com/watch?v=jY3tPM1oNyU (cit. on p. 221).

Weir, Paul (Mar. 3, 2017). *The Sound of No Man's Sky*. San Francisco, California, United States of America: Game Developers' Conference. URL: https://www.gdcvault.com/play/1024067/The-Sound-of-No-Man (cit. on p. 43).

Appendices

A

Platforms Used in Ludography

Game	Platform Used
.kkrieger (.theprodukkt 2004)	PC
Abzû (Giant Squid Studios 2016)	PC, via Epic Games Launcher
Ape Out (Cuzzillo 2019)	Referenced only
Apotheon (Alientrap 2015)	Referenced only
Asheron's Call (Turbine Entertainment Software 1999)	Referenced only
Audiosurf (Fitterer 2008)	PC, via Steam

Table A.1 – continued from previous page

Game	Platform Used
Ballblazer (Lucasfilm Games 1984)	Emulation, played on PC via VICE[1]
Banjo-Kazooie (Rare 1998)	Emulation, played on PC via Project64[2]
Bastion (Supergiant Games 2011)	PC, via Steam
Batman: Arkham Knight (Rocksteady Studios 2015)	Referenced only
Bioshock (2K Boston and 2K Australia 2011)	PC, via Steam
Borderlands (Gearbox Software 2009)	PC, GOTY version via Steam
Cadence of Hyrule (Brace Yourself Games 2019)	Referenced only
Call of Duty (Infinity Ward 2003)	Referenced only
Cavelon (Jetsoft 1983)	Emulation, played on PC via MAME[3]
Civilization IV (Firaxis Games 2005)	PC, via Steam
Colossal Cave Adventure (Crowther and Woods 1977)	Emulation, played online via People's Republic of Interactive Fiction (`http://pr-if.org/play../`)
Crypt of the NecroDancer (Brace Yourself Games 2015)	PC, via Steam
Cuphead (StudioMDHR 2017)	PC, via Steam

[1] VICE can be used to emulate games made for the Commodore 64, and can be found at `http://vice-emu.sourceforge.net/`.
[2] Project64 can be used to emulate games made for the Nintendo 64, and can be found at `https://www.pj64-emu.com/`.
[3] MAME can be used to emulate games on a variety of different platforms, and can be found at `https://www.mamedev.org/`.

Table A.1 – continued from previous page

Game	Platform Used
Dance Dance Revolution (Konami 1997)	Played at a physical arcade in Malta
Dark Chronicle (Level-5 2002)	PlayStation 2
DOOM (id Software 2016)	Referenced only
Dragon Age: Inquisition (BioWare 2014)	Referenced only
Dropmix (Harmonix 2017)	Referenced only
Dune II (Westwood Studios 1992)	Referenced only
Dungeons and Dragons (Gygax and Arneson 1974)	4th and 5th editions of the game played as both player and dungeon master
Dwarf Fortress (Adams 2006)	PC
Dynamite Headdy (Treasure 1994)	Emulation, played on PC via Kega Fusion[4]
Electroplankton (Indieszero 2005)	Referenced only
Elite (Braben and Bell 1984)	Referenced only
Escape Point (Prechtl 2015)	PC
Final Fantasy VIII (Square 1999)	PlayStation
Final Fantasy X (Square 2001)	PlayStation 2
Fortnite: Battle Royale (Epic Games 2017)	Referenced only
Frogger (Konami 1981)	Emulation, played on PC via MAME
Gran Turismo (Polys Entertainment 1997)	Referenced only

[4]Kega Fusion can be used to emulate games made for the following consoles: Sega SG1000, SC3000, SF7000, Master System, Game Gear, Genesis/Megadrive, SVP, Pico, SegaCD/MegaCD and 32X, and can be found at `https://www.carpeludum.com/kega-fusion/`.

Table A.1 – continued from previous page

Game	Platform Used
Grand Theft Auto V (Rockstar North 2013)	Referenced only
Grim Fandango (LucasArts 1998)	PC, remastered version via Steam
Guitar Hero (Harmonix 2005)	Referenced only
Hotline Miami (Dennaton Games 2012)	PC, via Steam
Journey (thatgamecompany 2012)	PlayStation 3
King's Quest 6 (Sierra On-Line 1992)	PC
Kingdom Come: Deliverance (Warhorse Studios 2018)	Referenced only
Mellan himmel och hav (Wieslander and Björk 2003)	Referenced only
Minecraft (Mojang 2011)	PC
Monkey Island 2: LeChuck's Revenge (LucasArts 1991)	PC, Special Edition via Steam
Myst (Cyan 1993)	PlayStation
Never Alone (*Kisima Inŋitchuŋa*) (Upper One Games 2014)	PC, via Steam
No Man's Sky (Hello Games 2016)	Referenced only
No One Lives Forever (Monolith Productions 2000)	Referenced only
Ōkami (Clover Studio 2006)	Referenced only
Otocky (SEDIC 1987)	Referenced only

Table A.1 – continued from previous page

Game	Platform Used
Pikmin (Nintendo EAD 2001)	Emulation, played on PC via Dolphin[5]
Pixelfield (Epic Stars Inc. 2016)	Referenced only
Pokémon Red (Game Freak 1996)	Emulation, played on PC via VisualBoy Advance[6]
Pong (Atari 1972)	Referenced only
Pop Tunes (Hoeberechts et al. 2012)	Referenced only
PlayerUnknown's Battlegrounds (PUBG Corporation 2017)	Referenced only
Resident Evil (Capcom 1996)	Referenced only
Resident Evil 7: Biohazard (Capcom 2017)	Referenced only
Rogue (Toy et al. 1980)	Emulation, played on PC via DOSBox[7]
Silent Hill 3 (Konami Computer Entertainment Tokyo 2003)	Referenced only
SingStar (SCEE Studio London 2004)	Referenced only
Space Channel 5 (United Game Artists 1999)	Referenced only

[5] Dolphin can be used to emulate games made for the Nintendo GameCube and Nintendo Wii, and can be found at `https://dolphin-emu.org/`.

[6] VisualBoy Advance can be used to emulate games made for the Nintendo Gameboy Advance, Nintendo Gameboy Color, and Nintendo Gameboy, and can be found at `http://www.emulator-zone.com/doc.php/gba/vboyadvance.html`.

[7] DOSBox can be used to emulate games built for a Microsoft DOS environment, and can be found at `https://www.dosbox.com/`.

Table A.1 – continued from previous page

Game	Platform Used
Space Invaders (Taito 1978)	Emulation, played on PC via Stella[8]
Spore (Maxis 2008)	PC, via Steam
Stardew Valley (Barone 2016)	PC, via Steam
Super Mario Bros. (Nintendo 1985)	Emulation, played on PC via Nestopia[9]
Superhot (Superhot Team 2016)	Referenced only
Technician Ted (Marsden and Cooke 1984)	Emulation, played on PC via Speccy[10]
The Curse of Monkey Island (LucasArts 1997)	PC
The Elder Scrolls IV: Oblivion (Bethesda Game Studios 2006)	PC
The Endless Forest (Tale of Tales 2006)	Referenced only
The Legend of Zelda: Breath of the Wild (Nintendo 2017)	Referenced only
The Legend of Zelda: The Ocarina of Time (Nintendo EAD 1998)	Emulation, played on PC via Project64
The Secret of Monkey Island (Lucasfilm Games 1990)	PC, Special Edition via Steam
The Witness (Thekla Inc. 2016)	PC, via Steam

[8]Stella can be used to emulate games built for the Atari 2600, and can be found at `https://stella-emu.github.io/`.
[9]Nestopia can be used to emulate games made for the Nintendo Entertainment Systems (NES), and can be found at `http://nestopia.sourceforge.net/`.
[10]Speccy can be used to emulate most models of the Sinclair ZX Spectrum, and can be found at `https://fms.komkon.org/Speccy/`.

Table A.1 – continued from previous page

Game	Platform Used
Thief: Deadly Shadows (Ion Storm 2004)	Referenced only
Uncharted: Drake's Fortune (Naughty Dog 2007)	Referenced only
Uurnog Uurnlimited (Nifflas Games 2017)	PC, via Steam
World of Warcraft (Blizzard Entertainment 2004)	PC, via Battle.net

B

Conventions of Mathematical Notation

B.1 Sets

Notation	Description
$\lvert S \rvert$	the cardinality of set S
2^S	the power set of S
$S \times S\prime$	the Cartesian product of S and $S\prime$
$\mathbb{Z}$	the set of all integers

Table B.1 – continued from previous page

Notation	Description
$\mathbb{Z}^+$	the set of all positive integers
$\mathbb{Z}^*$	the set of all non-negative integers
$\mathbb{R}$	the set of all real numbers
$\mathbb{Q}$	the set of all rational numbers

B.2 Symbols and Sequences

Notation	Description
$\perp$	the null symbol
e	an event
e^i	an event at position i of a sequence
e_i^j	a sequence of events indexed from i to j
ξ	the set of all possible events
ε	the empty sequence
$\mathcal{A}$	an alphabet of symbols
$\mathcal{A}^+$	the *positive closure* of $\mathcal{A}$ (the set of all non-empty sequences composed from elements of $\mathcal{A}$)
$\mathcal{A}^*$	the *Kleene closure*, defined as $\mathcal{A}^+$ (the set of all sequences composed from elements of $\mathcal{A}$, including ε)

Table B.2 – continued from previous page

Notation	Description
$e_i^j \in \mathcal{A}^*$	a sequence of symbols drawn from alphabet $\mathcal{A}$ and indexed from i to j, where $j \geq i \in \mathbb{Z}^+$
$\|$	sequence concatentation

B.3 Viewpoint Notation

Notation	Description
τ	a typed attribute
τ_i	a typed attribute, indexed i
τ_b	a typed basic attribute
τ_{b_i}	a typed basic attribute, indexed i
$[\tau]$	the syntactic domain of τ
$[\tau]\prime$	the syntactic domain of τ that has been seen by the model
$t \in [\tau]$	a viewpoint element
$\langle\tau\rangle$	the type set of τ
$[\![\tau]\!]$	the semantic domain of τ
$[\![\cdot]\!]_\tau : [\tau] \rightarrow [\![\tau]\!]$	semantic interpretation of $[\tau]$
$[\![\cdot]\!]\prime_\tau : [\![\tau]\!] \rightarrow [\tau]$	syntactic interpretation of $[\![\tau]\!]$
$\Psi_\tau : \xi^* \rightharpoonup [\tau]$	viewpoint function

Table B.3 – continued from previous page

Notation	Description
$\Phi_\tau : \xi^* \rightharpoonup [\tau]^*$	viewpoint matching function
$\Psi\prime_\tau : \xi^* \times [\tau] \rightarrow 2^{[\tau_b]}$	inverse viewpoint function
τ_l	a typed relational attribute
P	the domain of referent symbols
$\Upsilon_{\tau_l} : \xi^* \times \mathrm{P} \rightharpoonup [\tau_l]$	relational viewpoint function
$\Omega_{\tau_l} : \xi^* \times \mathrm{P}^* \rightharpoonup [\tau_l]^*$	matching function of relational viewpoint
$\Upsilon\prime_{\tau_l} : \xi^* \times [\tau_l] \times \mathrm{P} \rightarrow 2^{[\tau_b]}$	inverse relational viewpoint function

B.4 Probability and Information Theory

Notation	Description
$p(e\vert c)$ or $p(e^i\vert(e^i-1)_1$	probability of event e in a sequence given the context c

C
Glossary of Terms

abrupt cut transition A transition where the source piece is stopped abruptly and the target piece begins playing.. xiii, xxix, 60, 61, 63, 70, 76, 80, 91, 92, 96, 105, 111, 116, 120, 210, 211

battle royale A game genre where players must explore an in-game environment together with other players and survive for as long as possible in order to be the last person standing. Popular examples include *PlayerUnknown's Battlegrounds* (PUBG Corporation 2017) and *Fortnite: Battle Royale* (Epic Games 2017). 39

crossfading A transition where the source piece is faded out by lowering its volume, while at the same time the target piece is faded in by increasing its volume.. xiii, xiv, 2, 7, 60, 61, 65, 68, 70, 76, 89–91, 94, 96, 105, 111, 113–117, 120, 124, 163, 164, 188, 192, 197, 199, 200, 202–204, 207, 208, 210, 211, 213

diegetic Audio that occurs within the game itself and can therefore be heard by the player character.. 18, 19, 87, 88

extra-diegetic Audio that occurs outside the game and can not be heard by the player character.. 18, 19, 73, 87

first-person shooter A game genre where players control a character in first person view, i.e. the player's view of the world is through the eyes of a character in the game. Games in this genre tend to be action-heavy and revolve around the core mechanic of shooting at enemies. Popular examples include *Call of Duty* (Infinity Ward 2003) and *DOOM* (id Software 2016). xxxvii, 5

horizontal resequencing A scheduled music change based on in-game events that allow for the source music to resolve before the target piece begins playing.. xiii, xxviii, xxix, 54, 60, 63, 65, 67, 76, 80, 91–93, 96, 105, 111, 120, 124, 210, 211

inter-note transition Transitions between individual notes in a piece of music, often dealing with parameters for audio waveforms.. xxviii, 43, 45–47

inter-piece transition Transitions that connect two separate and distinct pieces of music.. xxviii, 42, 45, 47, 52–54, 58

inter-section transition Transitions between different sections in the same piece of music.. 42, 45, 47

platformer A game genre where players navigate a character through a level consisting of a series of platforms and other objects at various heights, while avoiding enemies. Popular examples include *Super Mario Bros.* (Nintendo 1985) and *Banjo-Kazooie* (Rare 1998). 23, 38, 64, 86, 171, 175

stinger A short piece of music (or a sound) that masks the change between one piece of music and another.. 60, 62, 67

trans-diegetic "[M]usic with no source in the game world but still [having] the ability to inform about events in that world" (Jørgensen 2007). 19

transition region A region that serves as a buffer zone between two areas in a game world, giving the music enough time to transition from one piece to another.. xiv, xxx, 123, 125, 126, 149–151, 153, 154, 160, 161, 176, 178, 179

vertical reorchestration The addition or removal of different musical elements or melody lines to the currently playing piece.. xiii, xxviii, 60, 63, 64, 67, 68, 124

viewpoint A lens or particular view that allows a sequence of musical events to be converted into a sequence of attributes taken from those events.. xiv, xxiv, xxx, 123, 129–143, 145–147, 149, 151, 153, 154, 156–158, 160, 164, 183–185, 212, 213, 215

voxel In a similar manner to how a pixel works in a 2D display, a voxel represents a value in a 3D grid. One popular implementation of voxels in computer games is in the sandbox game *Minecraft* (Mojang 2011). 35

D

Briefing Documents

D.1 Investigating the Detection of Musical Transitions

What is the aim of this research?

The purpose of this study is to investigate the success and suitability of different musical transition techniques. The study will focus particularly on how easily detectable different musical transitions are, and whether or not this affects how successful the transition is in the piece.

Who is conducting the research and who is it for?

Simon Cutajar is carrying out this research as part of his PhD studies, and is supervised by Robin Laney and Alistair Willis. Simon is a PhD student at the Open University and is part of the STEM faculty in the Computing and Communications department.

If I take part in this research, what will be involved?

Participants will be asked to conduct the study using our custom-designed software. The study will take approximately 40 minutes and would be conducted at the Open University, at a date and time that is convenient to you. To ensure your safety, all our researchers carry photographic identification. Participants are free to request a summary copy of the research findings if interested.

What will the study be like?

Participants will be evaluating 22 pieces of music and the transitions that occur in these pieces. To do so, participants will first listen to the piece in its entirety, without interaction. Once this is complete, participants will listen to the piece for the second time, clicking whenever they detect a musical transition within the piece. Participants will be able to delete erroneous detected transitions, but will not be able to add new detections after the second listening of the piece.

Participants will also be asked to rate each presented piece based on how well it transitions, the overall rate of change of the transition, how noticeable the transition was, and how well the transition fit with the rest of the piece. Participants are free to add further comments if they wish to do so.

Is it confidential?

Your participation will be treated in strict confidence in accordance with the Data Protection Act. No personal information will be passed to anyone outside the research team. We will write a report of the findings from this study, but no individual will be identifiable in published results of the research.

Participants are free to withdraw from the project without explanation or prejudice and are free to request the destruction of any gathered data until it is anonymised on April 1st 2018. After this point, the data will have been processed and it will not be possible to withdraw any provided unprocessed data.

All generated data will be stored on Open University servers and will be destroyed after 1 year. Any anonymised research data may be made available to other members of the research community for a period of 2 years, and if necessary, a pseudonym will be used when referring to any specific data.

D.2 Generative Transitions in a Game Environment

What is the aim of this research?

The purpose of this study is to compare a novel algorithm for generating musical transitions to an already existing musical transition technique. The study will focus on having participants evaluate musical transitions based on four different criteria, and evaluate each participant's level on immersion.

Who is conducting the research and who is it for?

Simon Cutajar is carrying out this research as part of his PhD studies, and is supervised by Robin Laney and Alistair Willis. Simon is a PhD student at the Open University and is part of the STEM faculty in the Computing and Communications department.

If I take part in this research, what will be involved?

Participants will be asked to conduct the study using our custom-designed software. The study will take approximately 30 minutes and would be conducted at the Open University, at a date and time that is convenient to you. To ensure your safety, all our researchers carry photographic identification. Participants are free to request a summary copy of the research findings if interested.

What will the study be like?

Participants will be evaluating 16 musical transitions between different pieces. To do so, participants must navigate through a game environment, travelling from one area to another in order to experience the musical transition. Once this is complete, participants will be presented with a screen allowing them to rate the musical transition based on the four presented criteria, as well as allowing them to rate the experience based on its level of immersion. Participants are free to add further comments if they wish to do so.

Is it confidential?

Your participation will be treated in strict confidence in accordance with the Data Protection Act. No personal information will be passed to anyone outside the research team. We will write a report of the findings from this study, but no individual will be identifiable in published results of the research.

Participants are free to withdraw from the project without explanation or prejudice and are free to request the destruction of any gathered data until it is anonymised on May 11th 2019. After this point, the data will have been processed and it will not be possible to withdraw any provided unprocessed data.

All generated data will be stored on Open University servers and will be destroyed after 1 year. Any anonymised research data may be made available to other members of the research community for a period of 2 years, and if necessary, a pseudonym will be used when referring to any specific data.

E

Asset Attribution

This chapter provides the necessary attribution for creators whose work was used to create the environment built for the study described in Chapter 6. Section E.1 lists attributions for any of the graphics used, Section E.2 lists attributions for any of the sound effects used, while Section E.3 lists the sheet music for the pieces used.

E.1 Graphics

Table E.1: Asset attribution for in-game models

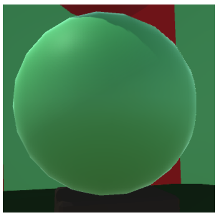
Asset Image	Attribution
	**Ball** **Environment:** Circus **Author:** Dmitriy Dryzhak (arvart-lit) **Website:** blendswap.com **Package:** Horror Circus Props (`https://www.cgtrader.com/3d-models/various/various-models/horror-circus-props`) **License:** Royalty Free
	**Balloons** **Environment:** Circus **Author:** Stephen Cutajar **Website:** **Package:** **License:** None

Table E.1 – continued from previous page

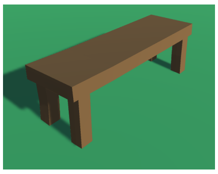
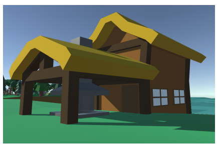
Asset Image	Attribution
	**Bench** **Environment:** Village **Author:** Matthias Pieroth (Jayanam Games) **Website:** patreon.com **Package:** Low Poly Nature `https://www.patreon.com/posts/free-low-poly-5-11361937` **License:** none
	**Blacksmith's Building** **Environment:** Village **Author:** Niko Abeler (H4kor) **Website:** itch.io **Package:** Low-Poly Village Buildings (`https://h4kor.itch.io/low-poly-village-buildings`) **License:** none

Table E.1 – continued from previous page

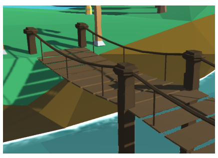
Asset Image	Attribution
	**Bone Pile** **Environment:** Spooky **Author:** 3D Sauce **Website:** sketchfab.com **Package:** `https://sketchfab.com/models/5ec580b41a934cea86d4297980d1378f` **License:** CC BY 4.0
	**Bridge** **Environment:** All **Author:** Matthias Pieroth (Jayanam Games), edited by Stephen Cutajar **Website:** patreon.com **Package:** Low Poly Nature `https://www.patreon.com/posts/free-low-poly-5-11361937` **License:** none

Table E.1 – continued from previous page

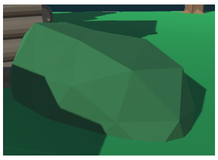
Asset Image	Attribution
	**Bush** **Environment:** Forest **Author:** Broken Vector **Website:** assetstore.unity.com **Package:** `https://t-allen-studios.itch.io/low-poly-log-cabin` **License:** none
	**Cabin - Forest** **Environment:** Forest **Author:** Tyler Allen (tallenstudios) **Website:** itch.io **Package:** Farm Animals (`https://vertexcat.itch.io/farm-animals-set`) **License:** CC0 1.0 Universal

Table E.1 – continued from previous page

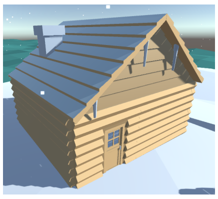
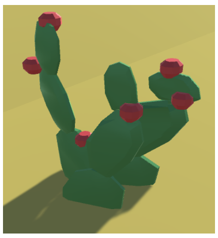
Asset Image	Attribution
	**Cabin - Snow** **Environment:** Snow **Author:** Kenney **Website:** kenney.nl **Package:** Holiday Kit (`https://kenney.nl/assets/holiday-kit`) **License:** CC0 1.0 Universal
	**Cactus** **Environment:** Desert **Author:** 23 Space Robots and Counting... **Website:** assetstore.unity.com **Package:** Low Poly Desert Pack (`https://assetstore.unity.com/packages/3d/environments/free-low-poly-desert-pack-106709`) **License:** None

Table E.1 – continued from previous page

Asset Image	Attribution
	**Cactus 2** **Environment:** Desert **Author:** Runemark Studio **Website:** assetstore.unity.com **Package:** POLYDesert (`https://assetstore.unity.com/packages/3d/environments/landscapes/polydesert-107196`) **License:** None
	**Cactus 3** **Environment:** Desert **Author:** Runemark Studio **Website:** assetstore.unity.com **Package:** POLYDesert (`https://assetstore.unity.com/packages/3d/environments/landscapes/polydesert-107196`) **License:** None

Table E.1 – continued from previous page

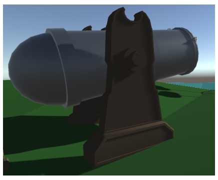
Asset Image	Attribution
	**Cannon** **Environment:** Circus **Author:** Dmitriy Dryzhak (arvart-lit) **Website:** blendswap.com **Package:** Horror Circus Props (`https://www.cgtrader.com/3d-models/various/various-models/horror-circus-props`) **License:** Royalty Free
	**Campfire** **Environment:** Forest, Spooky **Author:** Broken Vector **Website:** assetstore.unity.com **Package:** Low Poly Survival Essentials (`https://assetstore.unity.com/packages/3d/props/tools/low-poly-survival-essentials-109444`) **License:** None

Table E.1 – continued from previous page

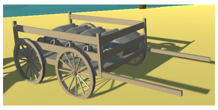
Asset Image	Attribution
	**Cart** **Environment:** Desert **Author:** Solum Night **Website:** assetstore.unity.com **Package:** Low Poly Pack - Environment Lite `https://assetstore.unity.com/packages/3d/props/exterior/low-poly-pack-environment-lite-102039` **License:** None

Table E.1 – continued from previous page

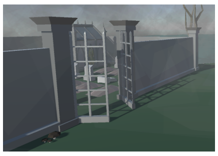
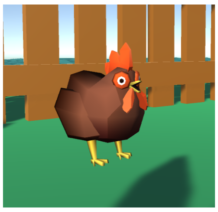
Asset Image	Attribution
	**Cemetery Walls & Gate** **Environment:** Spooky **Author:** Solum Night **Website:** assetstore.unity.com **Package:** Low Poly Pack - Environment Lite `https://assetstore.unity.com/packages/3d/props/exterior/low-poly-pack-environment-lite-102039` **License:** None
	**Chicken** **Environment:** Village **Author:** Ren Alea (vertexcat) **Website:** itch.io **Package:** Farm Animals (`https://vertexcat.itch.io/farm-animals-set`) **License:** CC0 1.0 Universal

Table E.1 – continued from previous page

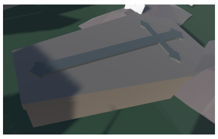
Asset Image	Attribution
	**Coffin** **Environment:** Spooky **Author:** Syoma Pozdeev (Addixon) **Website:** cgtrader.com **Package:** Scary Fantasy Halloween Pack (`https://www.cgtrader.com/free-3d-models/character/fantasy/free-demo-of-scary-fantasy-halloween-pack`) **License:** Free
	**Cow** **Environment:** Village **Author:** Ren Alea (vertexcat) **Website:** itch.io **Package:** Farm Animals (`https://vertexcat.itch.io/farm-animals-set`) **License:** CC0 1.0 Universal

Table E.1 – continued from previous page

Asset Image	Attribution
	**Crypt** **Environment:** Spooky **Author:** Syoma Pozdeev (Addixon) **Website:** cgtrader.com **Package:** Scary Fantasy Halloween Pack (`https://www.cgtrader.com/free-3d-models/character/fantasy/free-demo-of-scary-fantasy-halloween-pack`) **License:** Free

Table E.1 – continued from previous page

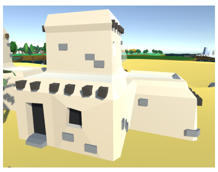
Asset Image	Attribution
	**Dead Tree** **Environment:** Spooky **Author:** Syoma Pozdeev (Addixon) **Website:** cgtrader.com **Package:** Scary Fantasy Halloween Pack (https://www.cgtrader.com/free-3d-models/character/fantasy/free-demo-of-scary-fantasy-halloween-pack) **License:** Free
	**Desert House** **Environment:** Desert **Author:** Caio Viero Baddo (CaioVB) **Website:** sketchfab.com **Package:** https://sketchfab.com/models/0d7c3e2ab66a4adfa72cb3e2065582f3 **License:** none

Table E.1 – continued from previous page

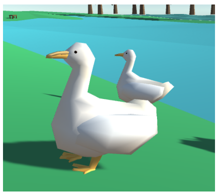
Asset Image	Attribution
	**Duck** **Environment:** Village **Author:** Ren Alea (vertexcat) **Website:** itch.io **Package:** Farm Animals (`https://vertexcat.itch.io/farm-animals-set`) **License:** CC0 1.0 Universal
	**Food Hut** **Environment:** Circus **Author:** Stephen Cutajar **Website:** **Package:** **License:** None

Table E.1 – continued from previous page

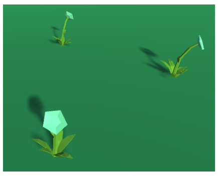
Asset Image	Attribution
	**Flowers** **Environment:** Forest **Author:** Rad-Coders **Website:** assetstore.unity.com **Package:** Low Poly: Foliage `https://assetstore.unity.com/packages/3d/vegetation/low-poly-foliage-66638` **License:** None
	**Flowers 2** **Environment:** Forest **Author:** Rad-Coders **Website:** assetstore.unity.com **Package:** Low Poly: Foliage `https://assetstore.unity.com/packages/3d/vegetation/low-poly-foliage-66638` **License:** None

Table E.1 – continued from previous page

Asset Image	Attribution
	**Forest Tree** **Environment:** Forest **Author:** Broken Vector **Website:** assetstore.unity.com **Package:** Low Poly Survival Essentials (`https://assetstore.unity.com/packages/3d/props/tools/low-poly-survival-essentials-109444`) **License:** None
	**Forest Tree 2** **Environment:** Forest **Author:** Works for Fun **Website:** assetstore.unity.com **Package:** Low Poly's Pack vol. 2 (`https://assetstore.unity.com/packages/3d/environments/low-poly-s-pack-vol-2-46375`) **License:** None

Table E.1 – continued from previous page

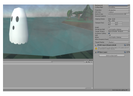
Asset Image	Attribution
	**Fox** **Environment:** Forest **Author:** PixelMannen **Website:** opengameart.org **Package:** `https://opengameart.org/content/fox-and-shiba` **License:** CC0 1.0 Universal
	**Ghost** **Environment:** Spooky **Author:** Stephen Cutajar **Website:** **Package:** **License:** None

Table E.1 – continued from previous page

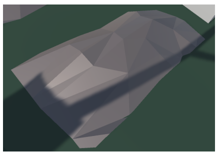
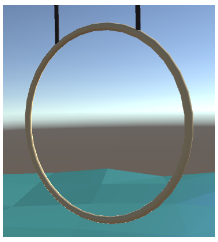
Asset Image	Attribution
	**Grave** **Environment:** Spooky **Author:** Syoma Pozdeev (Addixon) **Website:** cgtrader.com **Package:** Scary Fantasy Halloween Pack (`https://www.cgtrader.com/free-3d-models/character/fantasy/free-demo-of-scary-fantasy-halloween-pack`) **License:** Free
	**Hoop** **Environment:** Circus **Author:** Dmitriy Dryzhak (arvart-lit) **Website:** blendswap.com **Package:** Horror Circus Props (`https://www.cgtrader.com/3d-models/various/various-models/horror-circus-props`) **License:** Royalty Free

Table E.1 – continued from previous page

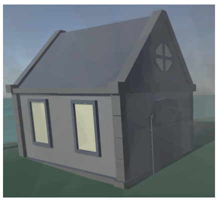
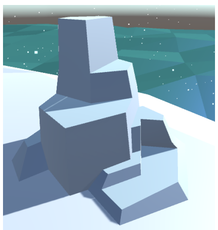
Asset Image	Attribution
	**House** **Environment:** Spooky **Author:** Solum Night **Website:** assetstore.unity.com **Package:** Low Poly Pack - Environment Lite `https://assetstore.unity.com/packages/3d/props/exterior/low-poly-pack-environment-lite-102039` **License:** None
	**Ice Rocks** **Environment:** Snow **Author:** Kenney **Website:** kenney.nl **Package:** Holiday Kit (`https://kenney.nl/assets/holiday-kit`) **License:** CC0 1.0 Universal

Table E.1 – continued from previous page

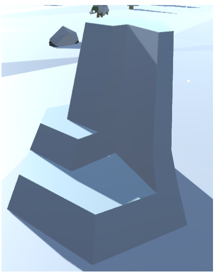
Asset Image	Attribution
	**Ice Rocks** **Environment:** Snow **Author:** Kenney **Website:** kenney.nl **Package:** Holiday Kit (`https://kenney.nl/assets/holiday-kit`) **License:** CC0 1.0 Universal
	**Ice Tree** **Environment:** Snow **Author:** Kenney **Website:** kenney.nl **Package:** Holiday Kit (`https://kenney.nl/assets/holiday-kit`) **License:** CC0 1.0 Universal

Table E.1 – continued from previous page

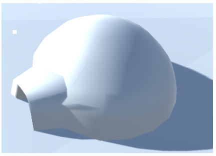
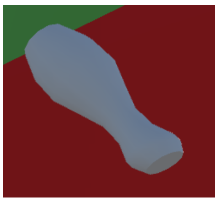
Asset Image	Attribution
	**Igloo** **Environment:** Snow **Author:** Stephen Cutajar **Website:** **Package:** **License:** None
	**Juggling Pin** **Environment:** Circus **Author:** Dmitriy Dryzhak (arvart-lit) **Website:** blendswap.com **Package:** Horror Circus Props (`https://www.cgtrader.com/3d-models/various/various-models/horror-circus-props`) **License:** Royalty Free

Table E.1 – continued from previous page

Asset Image	Attribution
	**Knife Throwing Hut** **Environment:** Circus **Author:** Stephen Cutajar **Website:** **Package:** **License:** None
	**Lighthouse** **Environment:** Snow **Author:** Dimmyxv **Website:** blendswap.com **Package:** Low-poly Iceberg, whale and penguins (`https://www.blendswap.com/blends/view/73565`) **License:** CC0 1.0 Universal

Table E.1 – continued from previous page

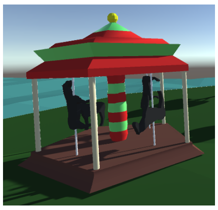
Asset Image	Attribution
	**Market Stall** **Environment:** Desert **Author:** Solum Night **Website:** assetstore.unity.com **Package:** Low Poly Pack - Environment Lite `https://assetstore.unity.com/packages/3d/props/exterior/low-poly-pack-environment-lite-102039` **License:** None
	**Merry-Go-Round** **Environment:** Circus **Author:** Stephen Cutajar **Website:** **Package:** **License:** None

Table E.1 – continued from previous page

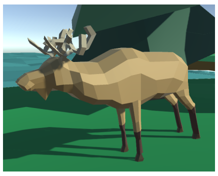
Asset Image	Attribution
	**Moose** **Environment:** Forest **Author:** m.altay **Website:** sketchfab.com **Package:** https://sketchfab.com/models/57f6633a35ab48849bf08676ac88d8a9 **License:** CC BY 4.0
	**Mushrooms** **Environment:** Forest **Author:** Rad-Coders **Website:** assetstore.unity.com **Package:** Low Poly: Foliage https://assetstore.unity.com/packages/3d/vegetation/low-poly-foliage-66638 **License:** None

Table E.1 – continued from previous page

Asset Image	Attribution
	**Palm Tree** **Environment:** Desert **Author:** 23 Space Robots and Counting... **Website:** assetstore.unity.com **Package:** Low Poly Desert Pack (`https://assetstore.unity.com/packages/3d/environments/free-low-poly-desert-pack-106709`) **License:** None
	**Penguin** **Environment:** Snow **Author:** Dimmyxv **Website:** blendswap.com **Package:** Low-poly Iceberg, whale and penguins (`https://www.blendswap.com/blends/view/73565`) **License:** CC0 1.0 Universal

Table E.1 – continued from previous page

Asset Image	Attribution
	**Plant** **Environment:** Forest **Author:** Rad-Coders **Website:** assetstore.unity.com **Package:** Low Poly: Foliage `https://assetstore.unity.com/packages/3d/vegetation/low-poly-foliage-66638` **License:** None
	**Plant 2** **Environment:** Forest **Author:** Rad-Coders **Website:** assetstore.unity.com **Package:** Low Poly: Foliage `https://assetstore.unity.com/packages/3d/vegetation/low-poly-foliage-66638` **License:** None

Table E.1 – continued from previous page

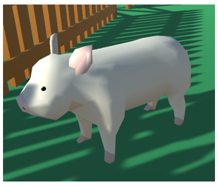
Asset Image	Attribution
	**Pig** **Environment:** Village **Author:** Ren Alea (vertexcat) **Website:** itch.io **Package:** Farm Animals (`https://vertexcat.itch.io/farm-animals-set`) **License:** CC0 1.0 Universal
	**Player Character** **Environment:** All **Author:** David Costine (omegagrim) **Website:** itch.io **Package:** Mages (`https://omegagrim.itch.io/lowpoly-mage-asset`) **License:** Free

Table E.1 – continued from previous page

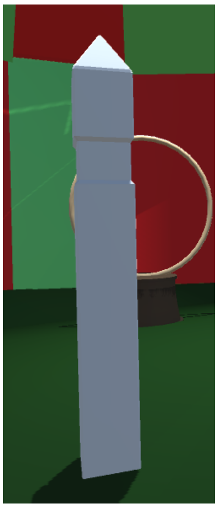
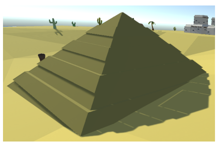
Asset Image	Attribution
	**Post** **Environment:** Circus **Author:** Dmitriy Dryzhak (arvart-lit) **Website:** blendswap.com **Package:** Horror Circus Props (`https://www.cgtrader.com/3d-models/various/various-models/horror-circus-props`) **License:** Royalty Free
	**Pyramid** **Environment:** Desert **Author:** Aquarius Max **Website:** assetstore.unity.com **Package:** `https://assetstore.unity.com/packages/3d/environments/fantasy/desert-sandbox-lite-25935` **License:** None

Table E.1 – continued from previous page

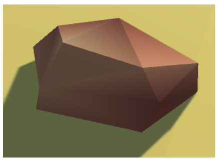
Asset Image	Attribution
	**Rock** **Environment:** Desert **Author:** 23 Space Robots and Counting... **Website:** assetstore.unity.com **Package:** Low Poly Desert Pack (`https://assetstore.unity.com/packages/3d/environments/free-low-poly-desert-pack-106709`) **License:** None
	**Seating Area** **Environment:** Circus **Author:** Stephen Cutajar **Website:** **Package:** **License:** None

Table E.1 – continued from previous page

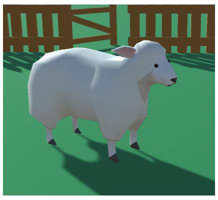
Asset Image	Attribution
	**Sheep** **Environment:** Village **Author:** Ren Alea (vertexcat) **Website:** itch.io **Package:** Farm Animals (`https://vertexcat.itch.io/farm-animals-set`) **License:** CC0 1.0 Universal
	**Shovel** **Environment:** Spooky **Author:** Syoma Pozdeev (Addixon) **Website:** cgtrader.com **Package:** Scary Fantasy Halloween Pack (`https://www.cgtrader.com/free-3d-models/character/fantasy/free-demo-of-scary-fantasy-halloween-pack`) **License:** Free

Table E.1 – continued from previous page

Asset Image	Attribution
	**Shrubs** **Environment:** Forest **Author:** Rad-Coders **Website:** assetstore.unity.com **Package:** Low Poly: Foliage `https://assetstore.unity.com/packages/3d/vegetation/low-poly-foliage-66638` **License:** None
	**Shrubs 2** **Environment:** Forest **Author:** Rad-Coders **Website:** assetstore.unity.com **Package:** Low Poly: Foliage `https://assetstore.unity.com/packages/3d/vegetation/low-poly-foliage-66638` **License:** None

Table E.1 – continued from previous page

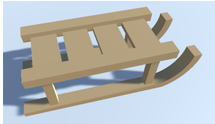
Asset Image	Attribution
	**Skull** **Environment:** Desert **Author:** 23 Space Robots and Counting... **Website:** assetstore.unity.com **Package:** Low Poly Desert Pack (`https://assetstore.unity.com/packages/3d/environments/free-low-poly-desert-pack-106709`) **License:** None
	**Sled** **Environment:** Snow **Author:** Kenney **Website:** kenney.nl **Package:** Holiday Kit (`https://kenney.nl/assets/holiday-kit`) **License:** CC0 1.0 Universal

Table E.1 – continued from previous page

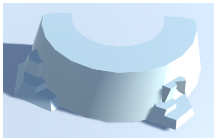
Asset Image	Attribution
	**Snow Fort** **Environment:** Snow **Author:** Kenney **Website:** kenney.nl **Package:** Holiday Kit (`https://kenney.nl/assets/holiday-kit`) **License:** CC0 1.0 Universal
	**Snowman** **Environment:** Snow **Author:** Kenney **Website:** kenney.nl **Package:** Holiday Kit (`https://kenney.nl/assets/holiday-kit`) **License:** CC0 1.0 Universal

Table E.1 – continued from previous page

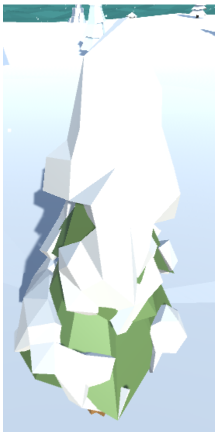
Asset Image	Attribution
	**Snow Tree** **Environment:** Snow **Author:** assetstore.unity.com **Website:** kenney.nl **Package:** `https://assetstore.unity.com/packages/3d/vegetation/trees/snowy-low-poly-trees-76796` **License:** None
	**Stand** **Environment:** Circus **Author:** Dmitriy Dryzhak (arvart-lit) **Website:** blendswap.com **Package:** Horror Circus Props (`https://www.cgtrader.com/3d-models/various/various-models/horror-circus-props`) **License:** Royalty Free

Table E.1 – continued from previous page

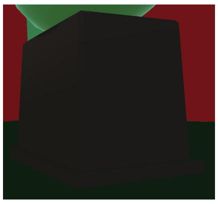
Asset Image	Attribution
	**Stand 2** **Environment:** Circus **Author:** Dmitriy Dryzhak (arvart-lit) **Website:** blendswap.com **Package:** Horror Circus Props (`https://www.cgtrader.com/3d-models/various/various-models/horror-circus-props`) **License:** Royalty Free
	**Statue - Snake** **Environment:** Desert **Author:** Aquarius Max **Website:** assetstore.unity.com **Package:** `https://assetstore.unity.com/packages/3d/environments/fantasy/desert-sandbox-lite-25935` **License:** None

Table E.1 – continued from previous page

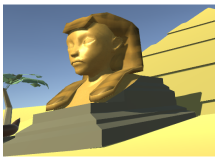
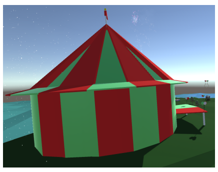
Asset Image	Attribution
	**Statue - Sphinx** **Environment:** Desert **Author:** Aquarius Max **Website:** assetstore.unity.com **Package:** `https://assetstore.unity.com/packages/3d/environments/fantasy/desert-sandbox-lite-25935` **License:** None
	**Tent - Circus** **Environment:** Circus **Author:** Stephen Cutajar **Website:** **Package:** **License:** None

Table E.1 – continued from previous page

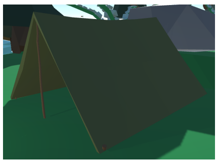
Asset Image	Attribution
	**Tent - Forest** **Environment:** Forest **Author:** Broken Vector **Website:** assetstore.unity.com **Package:** Low Poly Survival Essentials (`https://assetstore.unity.com/packages/3d/props/tools/low-poly-survival-essentials-109444`) **License:** None
	**Tombstone** **Environment:** Spooky **Author:** Syoma Pozdeev (Addixon) **Website:** cgtrader.com **Package:** Scary Fantasy Halloween Pack (`https://www.cgtrader.com/free-3d-models/character/fantasy/free-demo-of-scary-fantasy-halloween-pack`) **License:** Free

Table E.1 – continued from previous page

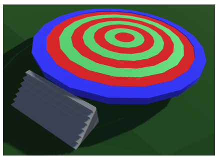
Asset Image	Attribution
	**Trampoline** **Environment:** Circus **Author:** Stephen Cutajar **Website:** **Package:** **License:** None
	**Tree** **Environment:** Forest **Author:** Taika Studios **Website:** itch.io **Package:** Low-Poly Ground (`https://taikastudios.itch.io/low-poly-ground`) **License:** none

Table E.1 – continued from previous page

Asset Image	Attribution
	**Tree** **Environment:** Village **Author:** Niko Abeler (H4kor) **Website:** itch.io **Package:** Low-Poly Village Buildings (`https://h4kor.itch.io/low-poly-village-buildings`) **License:** none
	**Tree Stump** **Environment:** Forest **Author:** Matthias Pieroth (Jayanam Games) **Website:** patreon.com **Package:** Low Poly Nature `https://www.patreon.com/posts/free-low-poly-5-11361937` **License:** none

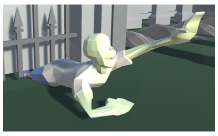
Table E.1 – continued from previous page

Asset Image	Attribution
	**Torch** **Environment:** Forest **Author:** MagicPot Inc. **Website:** assetstore.unity.com **Package:** LowPoly Dungeon Modules `https://assetstore.unity.com/packages/3d/environments/dungeons/lowpoly-dungeon-modules-108997` **License:** Free
	**Village House** **Environment:** Village **Author:** Niko Abeler (H4kor) **Website:** itch.io **Package:** Low-Poly Village Buildings (`https://h4kor.itch.io/low-poly-village-buildings`) **License:** none

Table E.1 – continued from previous page

Asset Image	Attribution
	**Well** **Environment:** Village **Author:** Niko Abeler (H4kor) **Website:** itch.io **Package:** Low-Poly Village Buildings (`https://h4kor.itch.io/low-poly-village-buildings`) **License:** none
	**Werewolf** **Environment:** Spooky **Author:** VitSh **Website:** sketchfab.com **Package:** `https://sketchfab.com/models/451d0af45af74892b119eabed444fa04` **License:** CC BY 4.0

Table E.1 – continued from previous page

Asset Image	Attribution
	**Villager** **Environment:** Forest **Author:** François Espagnet **Website:** sketchfab.com **Package:** `https://sketchfab.com/3d-models/the-tavern-man-1-c7732b90f63841d2898f5af22cce3423` **License:** CC BY 4.0
	**Zombie** **Environment:** Spooky **Author:** Tomás Laulhé (Quarternius) **Website:** reddit.com **Package:** `https://www.reddit.com/r/gamedev/comments/7pla8z/free_lowpoly_animated_zombie/` **License:** CC0 1.0 Universal

E.2 Sound Effects

Table E.2: Asset attribution for in-game sound effects

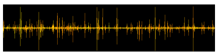
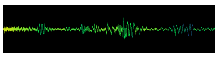
Sound Effect	Attribution
	**Campfire** **Environment:** Forest, Spooky **Author:** aerror **Website:** freesound.org **Package:** `https://freesound.org/people/aerror/sounds/350757/` **License:** CC0 1.0 Universal
	**Firework** **Environment:** Carnival **Author:** bmlake **Website:** freesound.org **Package:** `https://freesound.org/people/bmlake/sounds/251619/` **License:** CC BY 3.0

Table E.2 – continued from previous page

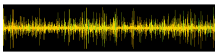
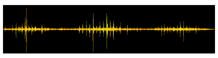
Sound Effect	Attribution
	**Footsteps: Bridge** **Environment:** All **Author:** Felix Blume (felix.blume) (edited by Simon Cutajar) **Website:** freesound.org **Package:** `https://freesound.org/people/felix.blume/sounds/397670` **License:** CC0 1.0 Universal
	**Footsteps: Desert** **Environment:** Desert **Author:** Cheeseheadburger (edited by Simon Cutajar) **Website:** freesound.org **Package:** `https://freesound.org/people/Cheeseheadburger/sounds/141517/` **License:** CC BY 3.0

Table E.2 – continued from previous page

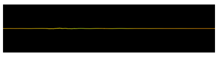
Sound Effect	Attribution
	**Footsteps: Grass, 1** **Environment:** Forest, Village, Carnival **Author:** Natalie Kirk, Owlish Media (OwlStorm) **Website:** freesound.org **Package:** Footsteps (`https://freesound.org/people/OwlStorm/sounds/151230/`) **License:** CC0 1.0 Universal
	**Footsteps: Grass, 2** **Environment:** Forest, Village, Carnival **Author:** Natalie Kirk, Owlish Media (OwlStorm) **Website:** freesound.org **Package:** Footsteps (`https://freesound.org/people/OwlStorm/sounds/151229/`) **License:** CC0 1.0 Universal

Table E.2 – continued from previous page

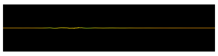
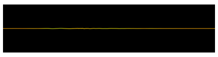
Sound Effect	Attribution
	**Footsteps: Grass, 3** **Environment:** Forest, Village, Carnival **Author:** Natalie Kirk, Owlish Media (OwlStorm) **Website:** freesound.org **Package:** Footsteps (`https://freesound.org/people/OwlStorm/sounds/151228/`) **License:** CC0 1.0 Universal
	**Footsteps: Grass, 4** **Environment:** Forest, Village, Carnival **Author:** Natalie Kirk, Owlish Media (OwlStorm) **Website:** freesound.org **Package:** Footsteps (`https://freesound.org/people/OwlStorm/sounds/151235/`) **License:** CC0 1.0 Universal

Table E.2 – continued from previous page

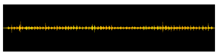
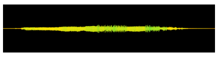
Sound Effect	Attribution
	**Footsteps: Snow** **Environment:** Snow **Author:** Jonathan Shaw (InspectorJ) (edited by Simon Cutajar) **Website:** freesound.org **Package:** `https://freesound.org/people/InspectorJ/sounds/421022` **License:** CC BY 3.0
	**Zombie, 1** **Environment:** Spooky **Author:** Evan X. Merz (PaulMorek) **Website:** freesound.org **Package:** Zombie Moans (`https://freesound.org/people/PaulMorek/sounds/196721/`) **License:** CC0 1.0 Universal

Table E.2 – continued from previous page

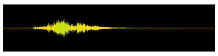
Sound Effect	Attribution
	**Zombie, 2** **Environment:** Spooky **Author:** Evan X. Merz (PaulMorek) **Website:** freesound.org **Package:** Zombie Moans (`https://freesound.org/people/PaulMorek/sounds/196720/`) **License:** CC0 1.0 Universal
	**Zombie, 3** **Environment:** Spooky **Author:** Evan X. Merz (PaulMorek) **Website:** freesound.org **Package:** Zombie Moans (`https://freesound.org/people/PaulMorek/sounds/196719/`) **License:** CC0 1.0 Universal

Table E.2 – continued from previous page

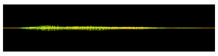
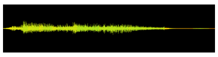
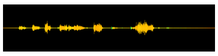
Sound Effect	Attribution
	**Zombie, 4** **Environment:** Spooky **Author:** Evan X. Merz (PaulMorek) **Website:** freesound.org **Package:** Zombie Moans (`https://freesound.org/people/PaulMorek/sounds/196718/`) **License:** CC0 1.0 Universal
	**Zombie, 5** **Environment:** Spooky **Author:** Evan X. Merz (PaulMorek) **Website:** freesound.org **Package:** Zombie Moans (`https://freesound.org/people/PaulMorek/sounds/196723/`) **License:** CC0 1.0 Universal

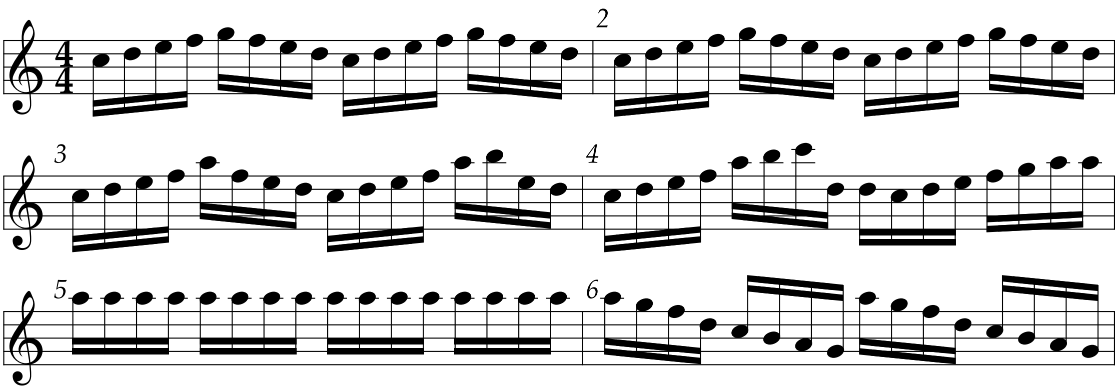
Table E.2 – continued from previous page

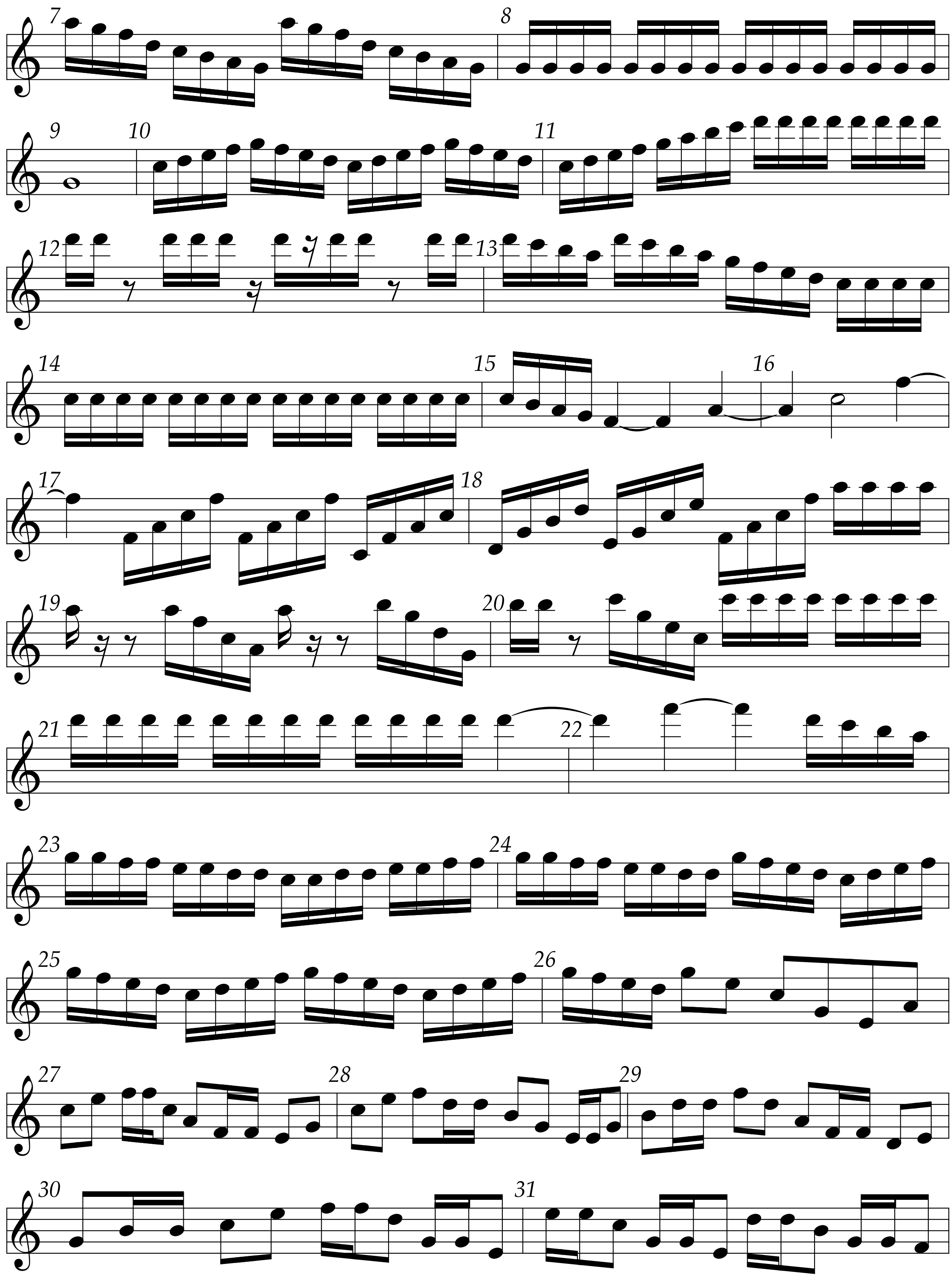
Sound Effect	Attribution
	**Zombie, 6** **Environment:** Spooky **Author:** Evan X. Merz (PaulMorek) **Website:** freesound.org **Package:** Zombie Moans (`https://freesound.org/people/PaulMorek/sounds/196722/`) **License:** CC0 1.0 Universal

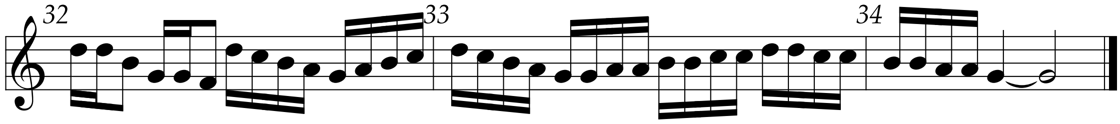
E.3 Music

E.3.1 Circus Island

E.3.1.1 Circus 1



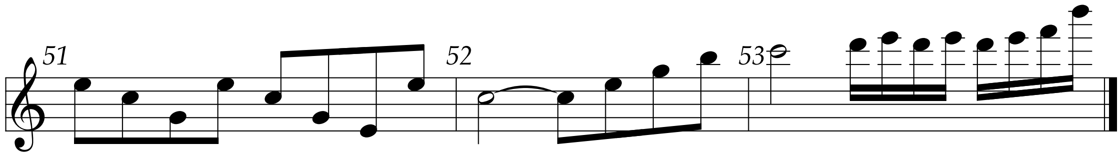




Example E.1: Circus 1

E.3.1.2 Circus 2

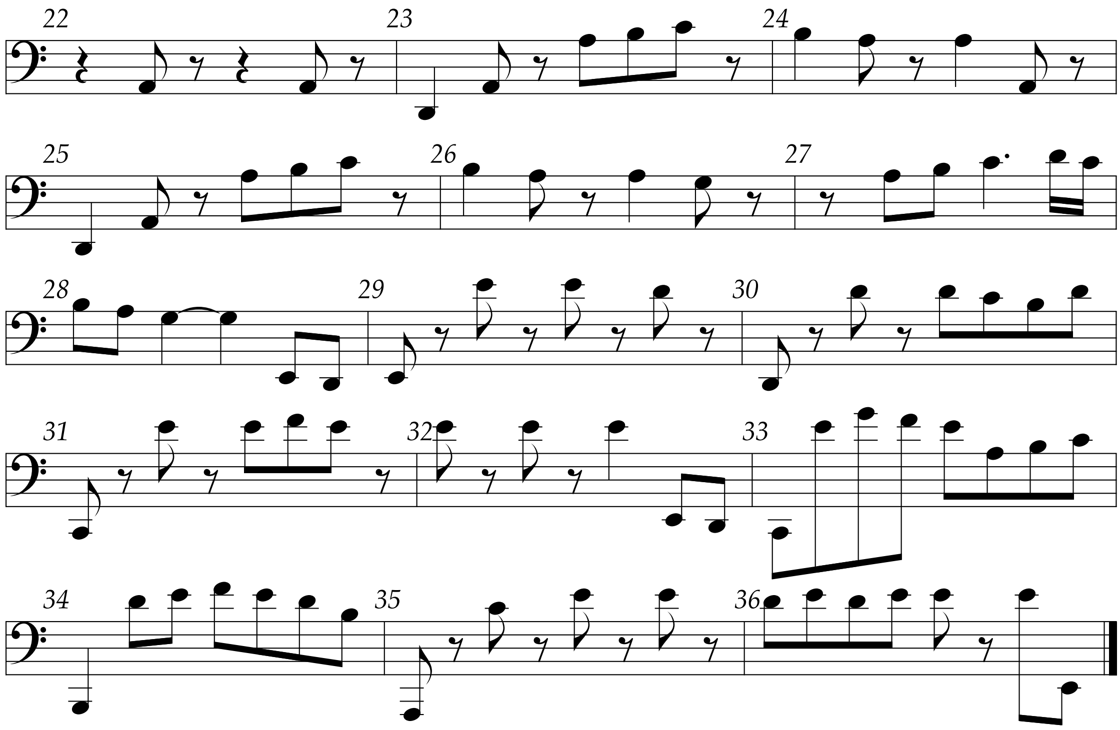

25 26 27
28 29
30 31
32 33 34
35 36
37 38 39
40 41 42
43 44 45
46 47
48 49 50



Example E.2: Circus 2

E.3.2 Desert Island

E.3.2.1 Desert 1



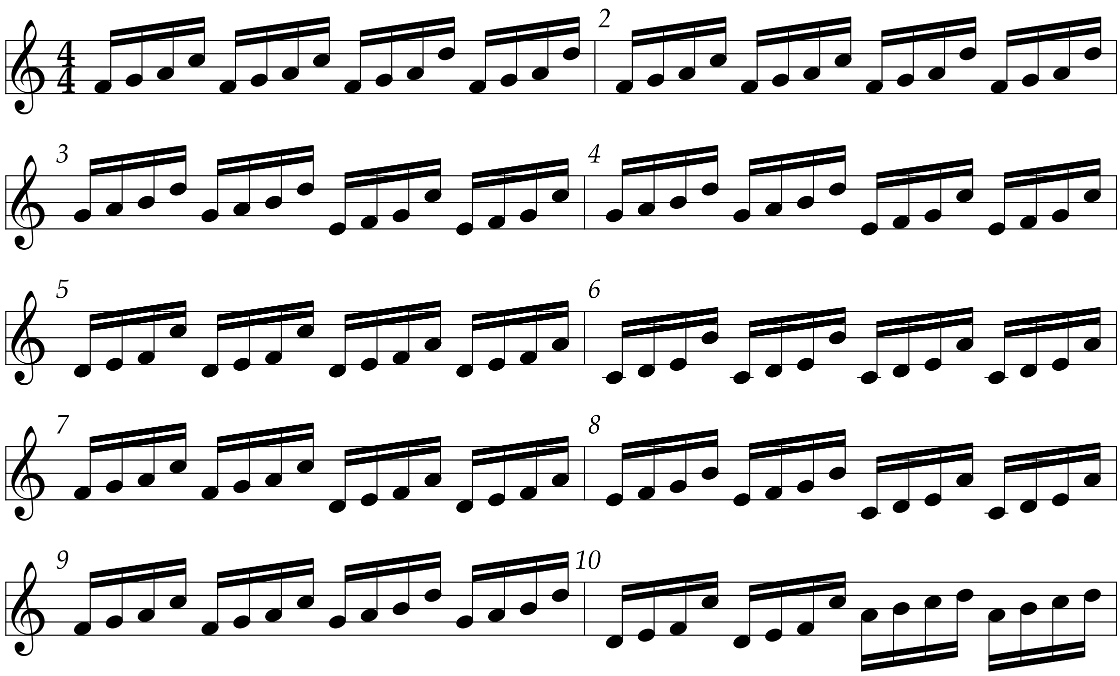
Example E.3: Desert 1

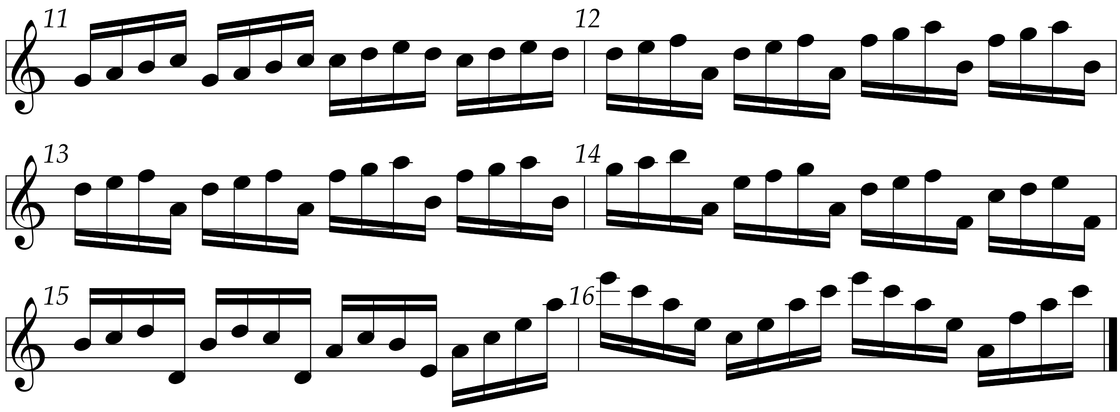
E.3.2.2 Desert 2

Example E.4: Desert 2

E.3.3 Forest Island

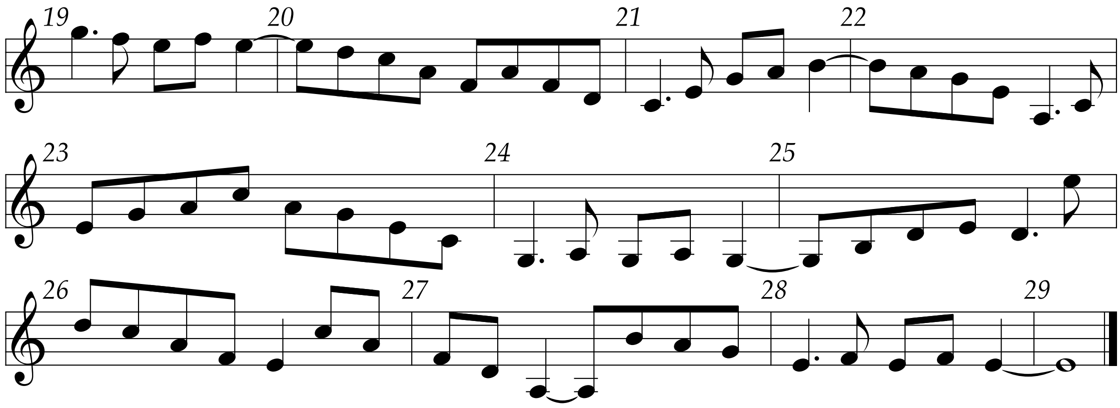
E.3.3.1 Forest 1





Example E.5: Forest 1

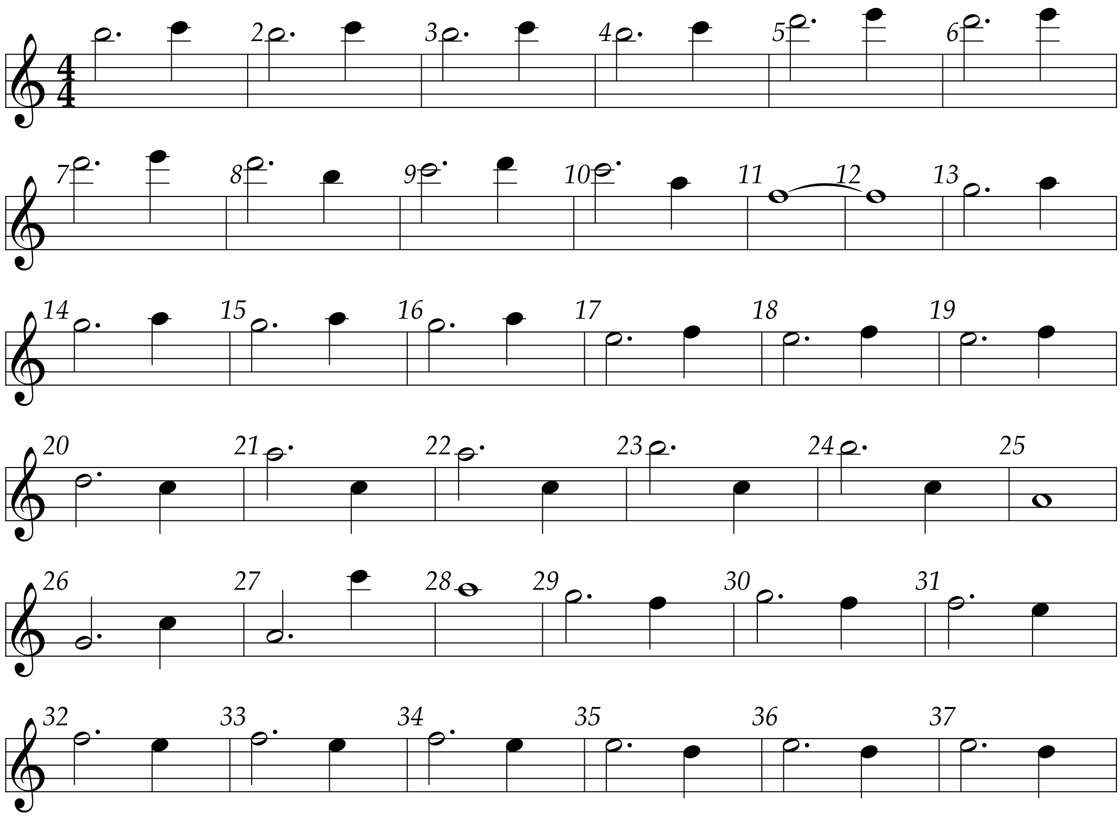
E.3.3.2 Forest 2

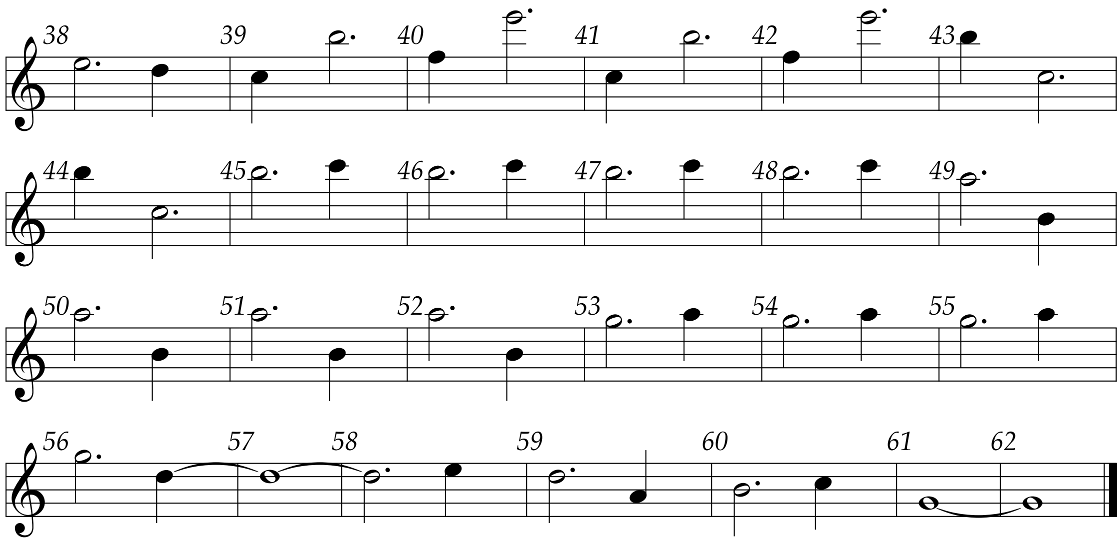


Example E.6: Forest 2

E.3.4 Ice Island

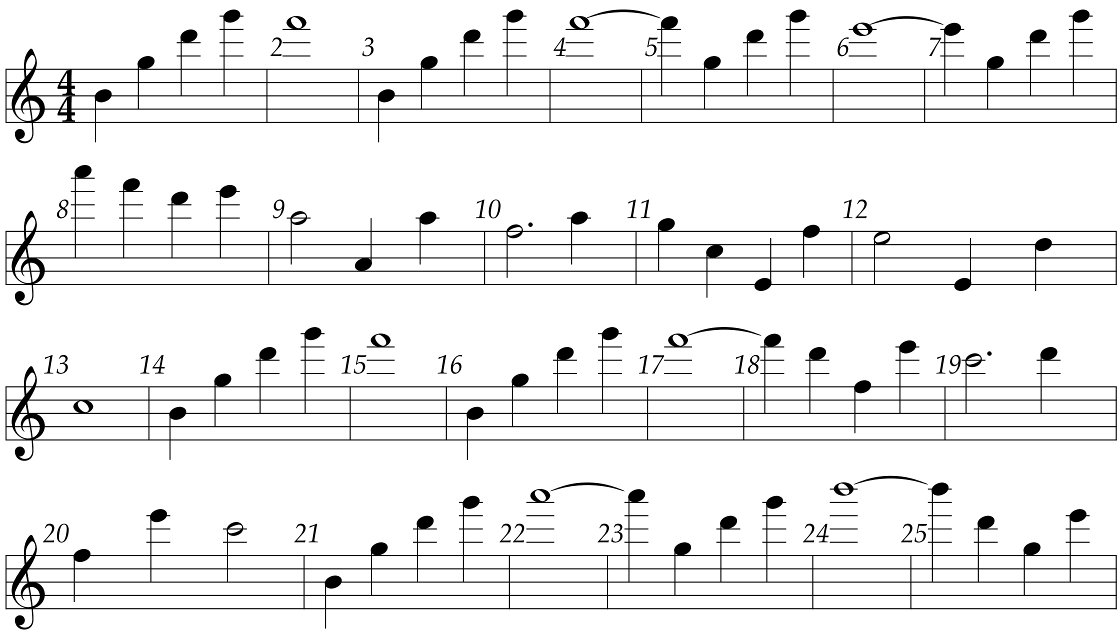
E.3.4.1 Ice 1

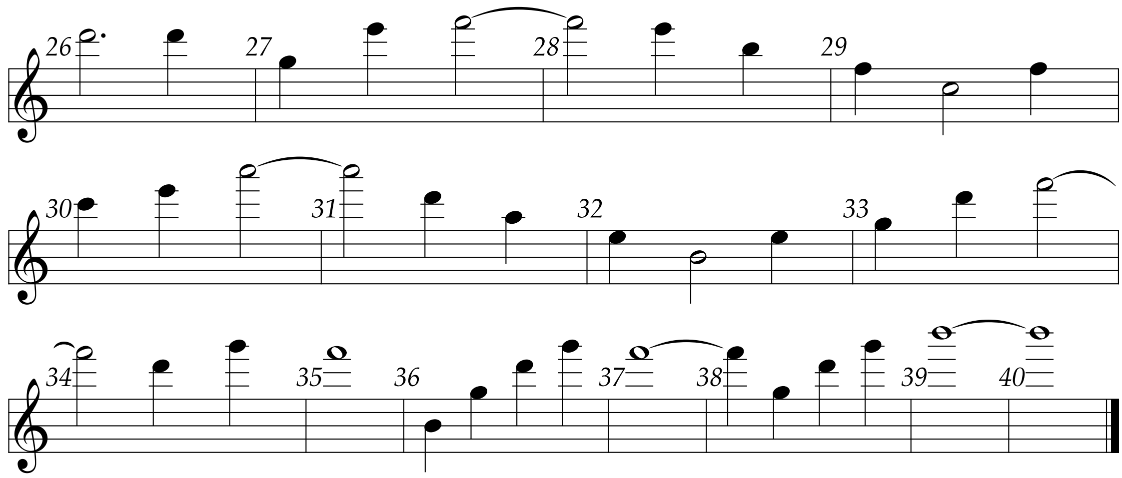




Example E.7: Ice 1

E.3.4.2 Ice 2

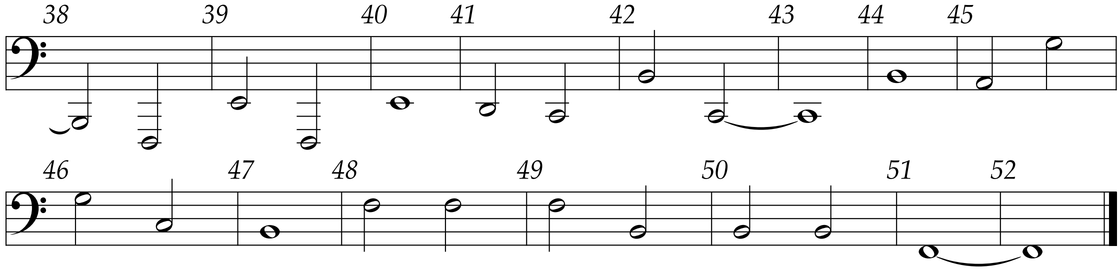




Example E.8: Ice 2

E.3.5 Spooky Island

E.3.5.1 Spooky 1



Example E.9: Spooky 1

E.3.5.2 Spooky 2

Example E.10: Spooky 2

E.3.6 Village Island

E.3.6.1 Village 1

Example E.11: Village 1

E.3.6.2 Village 2

Example E.12: Village 2

F

Timebase and **kern representation of musical event durations

Table F.1 shows a comparison of the representation of the duration of musical events using an appropriate timebase representation from M. Pearce 2005, p. 63 (with a granularity of 96), and **kern representation from Huron 1998.

Note Name	Timebase Value	**kern Value

Continued on next page

Table F.1 – *Continued from previous page*

	Note Name	Timebase Value	**kern Value
𝅝	Semibreve	96	1
𝅗𝅥..	Double Dotted Minim	84	2..
𝅗𝅥.	Dotted Minim	72	2.
𝅗𝅥	Minim	48	2
♩..	Double Dotted Crotchet	42	4..
♩. 1 of 3	Dotted Crotchet	36	4.
𝅗𝅥	Minim Triplet	32	3
♩	Crotchet	24	4
♪..	Double Dotted Quaver	21	8..
♪. 1 of 3	Dotted Quaver	18	8.
♩	Crotchet Triplet	16	6
♪	Quaver	12	8

Continued on next page

Table F.1 – *Continued from previous page*

	Note Name	Timebase Value	**kern Value
	Dotted Semiquaver	9	16.
	Quaver Triplet	8	12
	Semiquaver	6	16
	Dotted Demisemiquaver	4.5	32.
	Semiquaver Triplet	4	24
	Demisemiquaver	3	32
	Hemidemisemiquaver	1.5	64

Table F.1: A comparison of the representation of the duration of musical events using an appropriate timebase representation